GP4020 GPS Baseband Processor Design Manual

Publication Number: DM5280

Issue: 3
Revision: 002
Issued: January 2002

Zarlink Semiconductor, Cheney Manor
Swindon, Wiltshire, United Kingdom, SN2 2QW

Manual Revision History

Version	Revision	Date	Update Summary
1	001	February 2000	First Version
2	001	August 2000	GND and VDD pins marked as type “PWR” in tables 2.2 and 20.1. Modified TESTMODE (pin 74) definition. Removal of extra "™" and "®" trade-markings throughout. New BSIO Introduction (Secs 6.1, 6.2) Extensive DMAC usage procedures added (Sec 8) and section 17.4 deleted. Updated MPC Configuration for Memory Area 3 (Sec 11) New Note 1 added in Section 14.6.2 Updated Address Map info in Section 19. IO Cell DC Characteristics added to Section 20.
3	001	November 2001	Change of company identity from Mitel Semiconductor to Zarlink Semiconductor throughout. Revised values for UTC Error Budget figures (Sec 15.3). Figures 1.2 and 14.2 amended. Section 14.3.1 amended. "Trademarks" and "Document Conventions" sections added. Linked Cross-references to Sections, Figures and Tables added throughout.
3	002	January 2002	Document reformatted to US "Letter" size. Now 214 pages instead of ~300. Page numbers reformatted to show continuous page-numbering. Fig 14.4 updated to show revised TCXO connection to RF Front-end device.

TECHNICAL DOCUMENTATION - NOT FOR RESALE, PRINTED IN THE UNITED KINGDOM.

DOCUMENT NUMBER: **DM5280** 003 January 2002

Contents

Page

Related Products and Documents

Parameter	Description	Publication Reference
GP2015	GPS Receiver RF Front End Datasheet	DS4374
GP2010	GPS Receiver RF Front End Datasheet	DS4056
GP4020	GPS Baseband Processor Datasheet	DS5134
FIREFLY MF1	Microcontroller Core Design Manual	DM5003
	GPS Receiver Hardware Design Application Note	AN4855

Trademarks

Document References

References to the following documents are made within the GP4020 GPS Baseband Processor Design Manual:

1) "ARM7TDMI Technical Reference Manual"
ARM DDI 0029F, Rev 4 Copyright © ARM Limited 2001.
Arm Ltd. Documentation website (http://www.arm.com/arm/documentation?OpenDocument)

Document Conventions

The following terms which appear in the Manual, are defined here:

a) External device: device such as a memory or logic;

b) External Master: A Master device sited on the system bus;

c) low *or* clear: refers to a logical condition 0 of a signal or bit-field;

d) high *or* set: refers to a logical condition 1 of a signal or bit-field;

e) Numbers prefixed with '0x': hexadecimal;

f) Numbers prefixed with '0y': binary;

g) 'Reserved': When associated with a register field, the location should not be written to or read from. When used in a bit-field among other referenced fields, the default value must be maintained during write operations.

h) Register field bit positions are represented within square brackets, thus: [n] ;

1 INTRODUCTION

1.1 GP4020 GPS Baseband Processor Overview

This design manual describes the GP4020 GPS Baseband Processor, which is based on the Zarlink Semiconductor Firefly MF1 Microcontroller Core (ref. Firefly MF1 Core Design Manual (DM5003)), and a custom Navstar GPS C/A code 12-channel spread-spectrum correlator.

The GP4020 is a complete digital baseband processor for a Global Positioning System (GPS) receiver. It combines the 12-channel correlator function of the GP2021 with an advanced ARM7TDMI™ (Thumb®) microprocessor to achieve a higher level of integration, reduced system cost, reduced power consumption and added functionality. The GP4020 complements the GP2015 and GP2010 C/A code RF down-converters available from Zarlink Semiconductor.

The correlator section contains 12 identical tracking module blocks, one for each channel. Each channel contains all the components necessary for acquiring and tracking the received signal, and contains other functional blocks, which are used to produce part of the measurement data set. Individual channels may be deactivated for systems not requiring full 12-channel operation and thus allowing for reduced power consumption and processor loading.

The microprocessor section contains the Firefly MF1 micro-controller core, which includes an ARM7TDMI with a Thumb instruction de-compressor plus the Firefly BμILD module. Also included are a second UART, BμILD Serial I/O, General I/O and WATCHDOG functions.

1.2 Features

- Complete GPS correlator and Firefly MF1 micro-controller core
- ARM7TDMI (Thumb) microprocessor, with JTAG ICEBreaker™ debug interface
- Fully configurable external data-bus
- 12 Fully Independent Correlation Channels
- Low Voltage operation; 3.3V
- Low Current Power–Down Mode
- 1PPS UTC Aligned Timing Output, with 25ns resolution
- System Clock Generator with Phase Locked Loop, capable of producing Flexible microprocessor clock speeds
- 32KHz Real Time Clock
- Dual UART
- 3-wire BμILD Serial Input / Output (BSIO) interface
- 8 General Purpose Input / Output (GPIO) lines
- Boot ROM, allowing software upload via UART
- 8k Bytes internal SRAM
- Compatible with GP2015 and GP2010 RF Front Ends

1.3 Functional Description

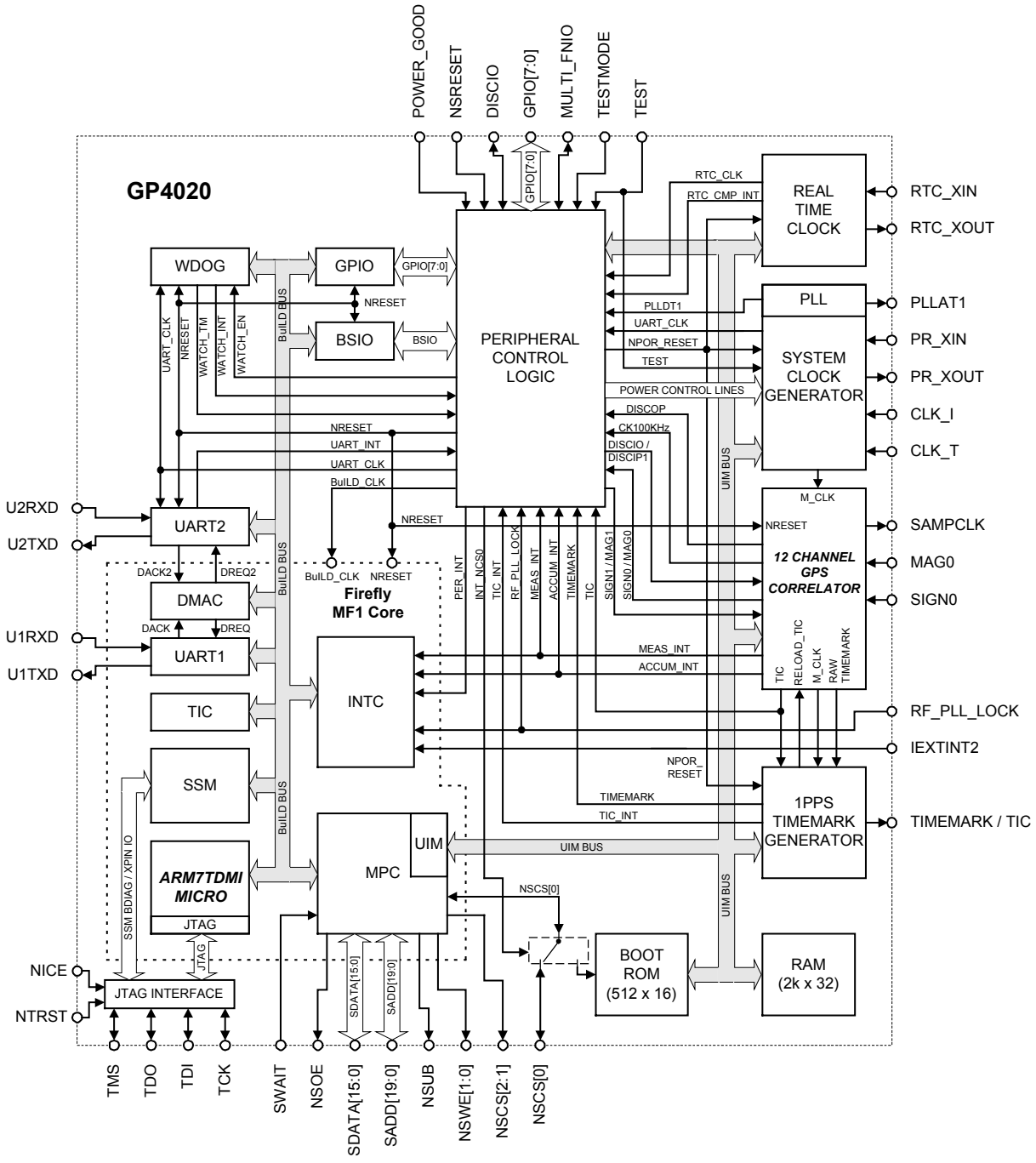


Figure 1.1 GP4020 Block Diagram

The GP4020 is a complete baseband processor for Navstar GPS C/A code signals. It incorporates a 12-channel GPS correlator, a Zarlink Firefly MF1 micro-controller core (incorporating the ARM7TDMI Thumb microprocessor), Real time Clock, 8k bytes of on-chip SRAM and a boot ROM. The GP4020 uses a fully configurable memory interface, allowing the use of both 8-bit and 16-bit external memory. A Block Diagram of the GP4020 appears in *Figure 1.1 GP4020 Block Diagram" above*.

1.3.1 ARM® Processor (ARM7TDMI)

The ARM7TDMI is a 32-bit RISC microprocessor core designed by Advanced RISC Machines (ARM). It uses a series 7 microprocessor Core, with the following functional extensions:

- Thumb (16-bit) instruction set
- Debug interface-using J-TAG.
- Fast Multiplier
- Embedded In-Circuit-Emulation capability

The ARM7TDMI is object-code compatible with all earlier ARM6 and ARM7 based products. The ARM7TDMI is a fully static design and as such consumes dynamic power only when clocked.

Details on the ARM7TDMI can be found in:

a) *Section 3 "ARM7TDMI MICROPROCESSOR" on page 19* of this manual

b) *Firefly MF1 Core Design Manual, (DM5003),* also available from Zarlink Semiconductor

c) *ARM7TDMI Technical Reference Manual (document reference ARM DDI 0029F),* which is downloadable (1.7 MB PDF) from ARM's website http://www.arm.com. The documentation download page can be found at: http://www.arm.com/arm/documentation?OpenDocument .

1.3.2 Boot ROM

The GP4020 BOOT ROM contains code, which is executed every time there is a complete system reset (i.e. when main power has been removed from the GP4020).

The code installed on the BOOT ROM, allows the GP4020 to undertake either of 2 functions after a complete reset:

- Run External FLASH EPROM from EPROM base address user to either run code direct from an external FLASH EPROM memory
- Load into the internal SRAM a unique program via the UART1 input. This could be used for test purposes, although the target use of this facility is to allow for field-upgrades of GPS receiver firmware, in conjunction with a FLASH EPROM.

Details can be found in *section 4 "BOOT ROM" on page 27*:

1.3.3 BµILD Bus

This is a modular bus architecture and specification, via which all on-chip modules communicate with each other. These modules can either be bus masters or slaves. A bus master can initiate a bus access, generate addresses and control read or write transfers. A bus slave responds to a bus master request when selected by the system address decoder, and may, if required, assert a wait signal on the bus until the relevant data transfer has been completed. All internal data transfers on the module bus are single cycle.

The Firefly MF1 micro-controller has three modules that are capable of operating as Bus masters. These are the ARM7TDMI Core, DMAC and SSM, described below.

1.3.4 BµILD Serial Input Output (BSIO)

This module produces a 2-channel 3-wire serial interface for upto 2 external "Slave" serial interface devices (e.g. serial EEPROM). It provides both MICROWIRE™ Interface and Serial Peripheral Interface (SPI™) compatibility.

Details can be found in *section 6 "BμILD SERIAL INPUT OUTPUT (BSIO) INTERFACE" on page 33.*

1.3.5 12 Channel Correlator (CORR)

This module contains 12 channels of PRN code correlators for spread-spectrum correlation of 12 simultaneous signals. Each channel contains an independent carrier DCO to allow independent mix down of a satellite signal to base-band before code correlation occurs. The correlator is designed to extract data modulated at a nominal chipping-rate of 1.023MBPS, and can be used on both Navstar C/A code GPS signals, and Inmarsat WAAS codes.

Details can be found in *section 7 "12-CHANNEL CORRELATOR (CORR)" on page 49*.

1.3.6 DMA Controller (DMAC)

Two DMA engines are available on the micro-controller. These are configured as a pair to provide a memory-to-memory DMA capability between UART1, UART2 and any location in the ARM7TDMI memory space. Alternatively, they may be used independently for fly-by transfers between the UARTs and either on-chip or off-chip locations. Single or multiple byte transfers (Demand or Burst Mode) are supported and may be word, half-word or byte wide.

Details can be found in *section 8 "DMA CONTROLLER (DMAC)" on page 91.*

1.3.7 Embedded Micro-Controller Debug Options

The Firefly MF1 Core incorporates three sophisticated methods of hardware and software debug. The options are:

- EmbeddedICE accessed via the ARM7TDMI JTAG interface.
- Angel Debug Monitor.
- Logic Analyser coupled with an Inverse Assembler accessed via the SSM debug interface.

The GP4020 can use any of these options, but special emphasis has been placed on the EmbeddedICE and Logic Analyser options. The JTAG and SSM debug interfaces are multiplexed onto the same pins, and can be selected by setting the NICE (pin 84) to High for SSM, or Low for JTAG.

1.3.8 Firefly MF1 Micro-Controller core

The Firefly MF1 Micro-controller is an Embedded Micro-controller core developed by Zarlink Semiconductor. It combines the processing power of the ARM7TDMI microprocessor with a number of peripheral components:

- Direct Memory Access Controller (DMAC)
- Interrupt Controller (INTC)
- Memory Peripheral Controller (MPC), incorporating Up-Integration Module (UIM)
- System Services Module (SSM)
- System Timer/Counter (SYSTIC)
- Universal Asynchronous Receiver / Transmitter (UART)

Details can be found in the *Firefly MF1 Core Design Manual, (DM5003),* also available from Zarlink Semiconductor.

1.3.9 General Purpose Input Output (GPIO)

This module provides eight I/O pins, which may be bit or byte addressed and configured in a latched or transparent mode. When in byte mode, buffer full/empty flags are available which can be used to generate an interrupt to the ARM7TDMI processor.

Details can be found in *section 9 "GENERAL PURPOSE INPUT OUTPUT (GPIO) INTERFACE" on page 103.*

1.3.10 Interrupt Controller (INTC)

The ARM7TDMI core accepts two types of interrupt: Normal (IRQ) and Fast (FIQ). All Interrupts can be switched between types, depending upon the relative priorities required.

The INTC is the central control logic that decodes the priority level and handles interrupt request signals from a number of external sources.

Details can be found in *section 10 "INTERRUPT CONTROLLER (INTC)" on page 107.*

1.3.11 Memory/Peripheral Controller (MPC)

The MPC ensures the correct multiplexing of data is applied for bus transfers between 8-, 16- or 32-bit on-chip macrocells, and 8- 0r 16-bit off-chip peripherals. Four different contiguous memory areas are available, each with an address range of one MByte, with individually programmable wait- and stop-state generation. A “SWAP” function allows memory area “1”, which is addressed at system reset, to be switched with memory area “4”. This allows, for example, booting from ROM and then switching memory area 1 to address SRAM so that time-critical software and interrupt routines can operate from fast memory.

Details can be found in *section 11 "MEMORY PERIPHERAL CONTROLLER (MPC)" on page 109.*

1.3.12 Peripheral Control Logic (PCL)

The GP4020 incorporates some specific control logic, which is used to control a number of functions:

- System Reset Control
- System Power-down, Sleep and Wake-up Control
- System Status and Control Registers
- Signal input/output multiplex control

Details can be found in *section 12 "PERIPHERAL CONTROL LOGIC (PCL)" on page 113.*

1.3.13 Internal SRAM

The GP4020 contains 8k bytes (configured as 2k x 32-bit) of high-speed (6ns) Static RAM. This can be used for either:

- Non-volatile storage of GPS data (Almanac, Ephemeris, Position and Receiver Clock Offset), while the receiver power is disabled.
- A High-speed Interrupt Service Routine, while the GP4020 is powered up.

The internal SRAM appears at GP4020 Base Address 0x6000 0000, served by the MPC Memory Area 4. An MPC SWAP function can swap this memory space with 0x0000 0000 if required.

Since the internal SRAM is high-speed, it can be accessed with Zero wait-states through the Memory Peripheral Controller. Refer to *section 11 "MEMORY PERIPHERAL CONTROLLER (MPC)" on page 109, for more information.*

1.3.14 Real Time Clock (RTC)

The GP4020 Real Time Clock uses an external 32kHz crystal to give an indication of time to the GP4020 chip, when the device is in Reset / Power Down. If a backup battery is included in a GPS receiver using the GP4020, the RTC will continue to operate regardless of the reset-state of the rest of the device.

The RTC is incremental, which means that the number of seconds from a reset point is accumulated. The Real Time Clock does NOT in itself contain a record of Gregorian Date, and so is NOT be affected by Year 2000 compliance issues.

Details can be found in *section 13 "REAL TIME CLOCK (RTC)" on page 131.*

1.3.15 System Clock Generator (SCG)

The GP4020 System Clock Generator is used to provide 2 system clocks:

- The M_CLK for the 12-channel Correlator; this is derived from the CLK_T and CLK_I inputs from the RF Front-end device and MUST be 40MHz. This clock is fundamental to the correlator function, and must be phase-locked to the RF Front-end

- The B_CLK for ALL components on the BμILD Bus; this can be derived from M_CLK (see above) in conjunction with a PLL and a divider to generate a wide range of clock frequencies. In this way, the B_CLK can be phase-locked to the RF Front-end. The clock can also be derived from an independent Crystal source.

Details can be found in *section 14 "SYSTEM CLOCK GENERATOR (SCG)" on page 135.*

1.3.16 System Services Module (SSM)

The System Services Module (SSM) ensures correct bus operation through a number of modes (reset, initialisation, debug, etc.). It provides diagnostic broadcast of address and data for internal transfers along with information about the current operating mode.

Additionally the SSM System Configuration Register controls the operating mode of the GP4020.

Specifically the System Services Module performs the following functions:

- Control the BμILD bus operational mode
- Arbitrate amongst competing resources for BμILD bus mastership
- Interface to external bus masters and manufacturing testers
- Respond to power-down requests from the Power Control logic within the Core Peripheral Control Logic block.
- Control the activities of all BμILD bus modules during system debug activity.
- Broadcast information about BμILD bus activity for external diagnostics
- Hold BμILD bus logic levels when no other bus-master is driving
- Register System Configuration data.

Further details of the function and programming System Services Module can be found in *Sections 2 and 8 of the "Firefly MF1 Core Design Manual" DM5003, available from Zarlink Semiconductor.*

1.3.17 System Timer/Counters (SYSTIC)

Two dual independent 32-bit timer/counters, with an 8-bit pre-scaler capability for each counter, are provided (Timers 1A, 1B, 2A and 2B). These are synchronous to the system clock and may be polled, or set-up to generate interrupts on over-run, with auto-reload.

The TIC functions provided by this module are part of the Firefly MF1 core. Timer 1 (TIC1) appears at GP4020 Base Address 0xE000 E000, and Timer 2 (TIC2) appears at Address 0xE000 F000. TIC enable (TEN) lines are not available externally on the GP4020, but are tied Low on- chip. The TIC functions can be made available by setting the External Enable Polarity bit of the TIC Control/Status register to logic “0”.

These timer / counters are NOT required by the GPS function in a GP4020 based GPS receiver. However, full programming details of the programming of the System Timer/Counter can be found in *Section 7 of the "Firefly MF1 Core Design Manual" DM5003, available from Zarlink Semiconductor.*

1.3.18 1PPS Timemark Generator (1PPS)

The GP4020 Timemark generator is used in conjunction with software to produce a 1 Pulse-Per-Second (1PPS) output pulse, which is aligned to Universal Time Co-ordinated (UTC) to a resolution of 25ns. The accuracy of time transmitted from the Navstar GPS space-segment is very high, and this can be used to provide a mobile timing reference to a similar accuracy.

Details can be found in *section 15 "1PPS TIMEMARK GENERATOR" on page 149.*

1.3.19 Up Integration Module (UIM)

This module provides a series of internal connection ports, which mimic the MPC external interface. This allows customer logic, which has been developed externally and accessed via the MPC interface, to be quickly and efficiently integrated to produce a complete ASIC.

1.3.20 Universal Asynchronous Receiver Transmitter (UART)

The full duplex asynchronous channel provides an RS232 type interface, which supports a XON/XOFF software protocol. The Receive and Transmit channels are double buffered. The UART may be either Polled, or use an interrupt scheme for module bus transfers. An internal Baud rate generator can provide selectable data rates, derived from on-chip sources for a Rx/Tx pair. Directly triggered DMA transfers with the UART are also possible without the need for CPU intervention.

Details can be found in *section 17 "UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)" on page 169.*

1.3.21 Watchdog (WDOG)

The GP4020 Watchdog can be used to detect hardware or software run-time errors, and reset the system. The processor is required to reset the watchdog periodically; failure to do so will result in a chip-wide reset.

Details can be found in *section 18 "WATCHDOG TIMER (WDOG)" on page 176.*

1.4 Typical Application

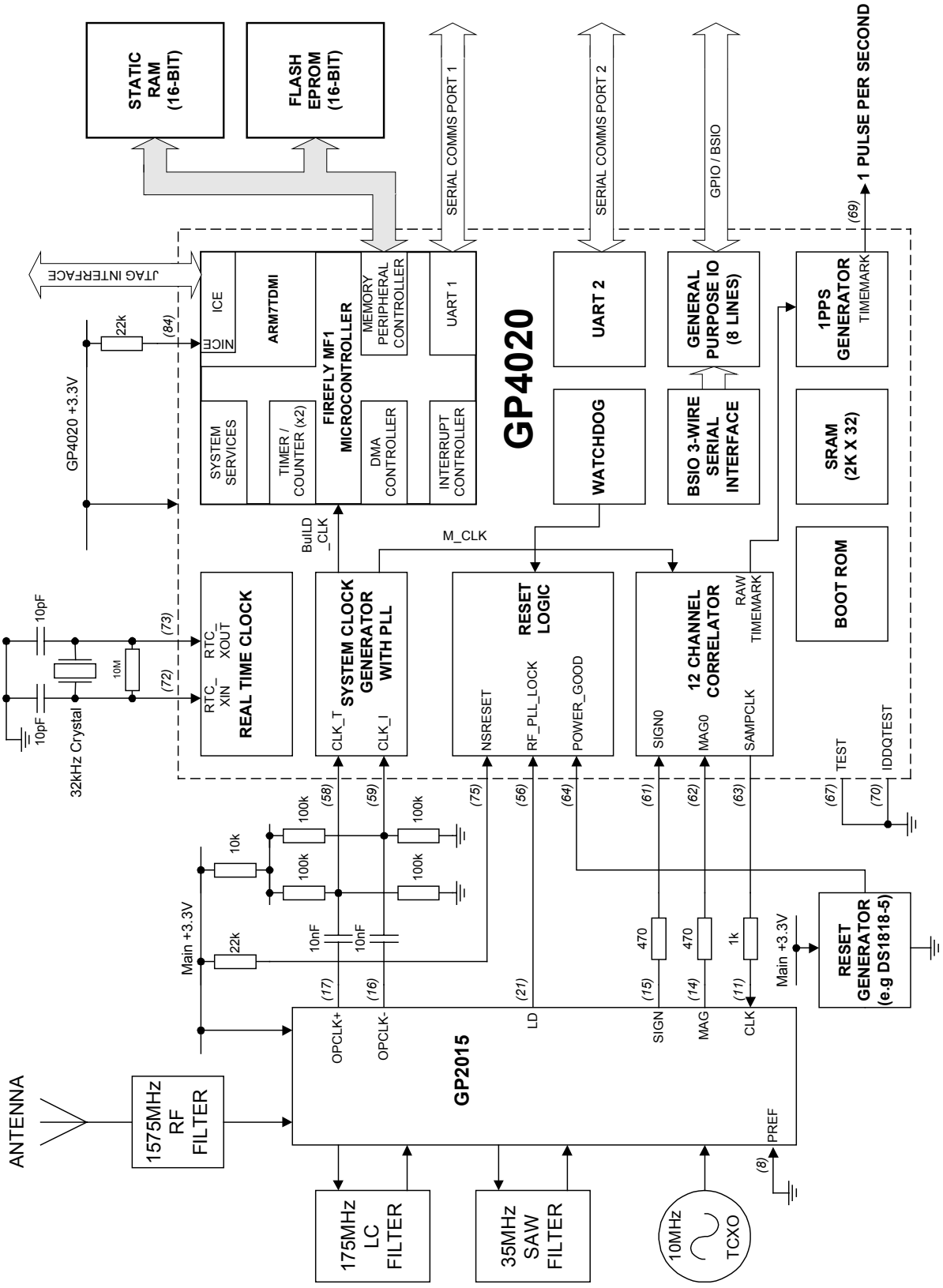


Figure 1.2 Block Diagram of typical GP4020 based GPS receiver

Figure 1.2 *above* shows a typical GPS receiver employing a GP2015 RF front–end, and a GP4020 correlator. The RF section, GP2015, performs down conversion of the L1 (1575.42MHz) signal for digital baseband processing. The resultant signal is then correlated in the GPS correlator within the GP4020 with an internally generated replica of the satellite PRN code to be received. Individual codes for each channel may be selected independently to enable acquisition and tracking of up to 12 different satellites simultaneously.

The results of the correlations form the accumulated data, which is transferred to the microprocessor to give the broadcast satellite data (the 'Navigation Message') and to control the software signal tracking loops.

Note that the GP4020 is designed to operate from an independent PSU supply, so that it can remain active while all the peripheral components are powered off. This is signified by the use of the PSU names "Main +3.3V", and "GP4020 +3.3V". Essentially, the GP4020 can be used with a battery supply while the Main +3.3V supply is disabled.

This page intentionally left blank.

2 GP4020 PACKAGE AND ELECTRICAL CONNECTIONS

2.1 GP4020 100-pin Package Dimensions

The GP4020 GPS Baseband Processor is available from Zarlink Semiconductor in a 100-pin gull-wing Thin Quad Flat Package (TQFP). Ordering information for the GP4020 are shown in the “GP4020 GPS Baseband Processor Datasheet” DS5134, available from Zarlink Semiconductor.

Figure 2.1 below shows the pin distribution around the package. *Figure 2.2 on page 12* shows the default package outline drawing. *Table 2.1 on page 13* gives values for each of the package dimensions.

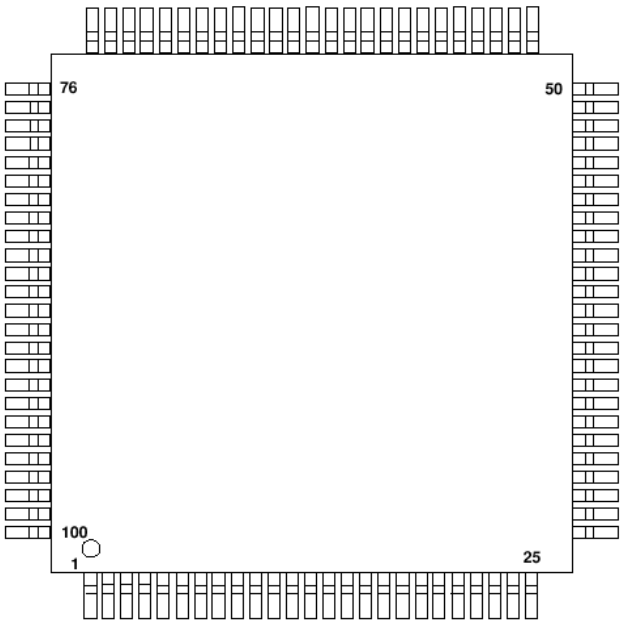


Figure 2.1 GP4020 100-pin package pin distribution

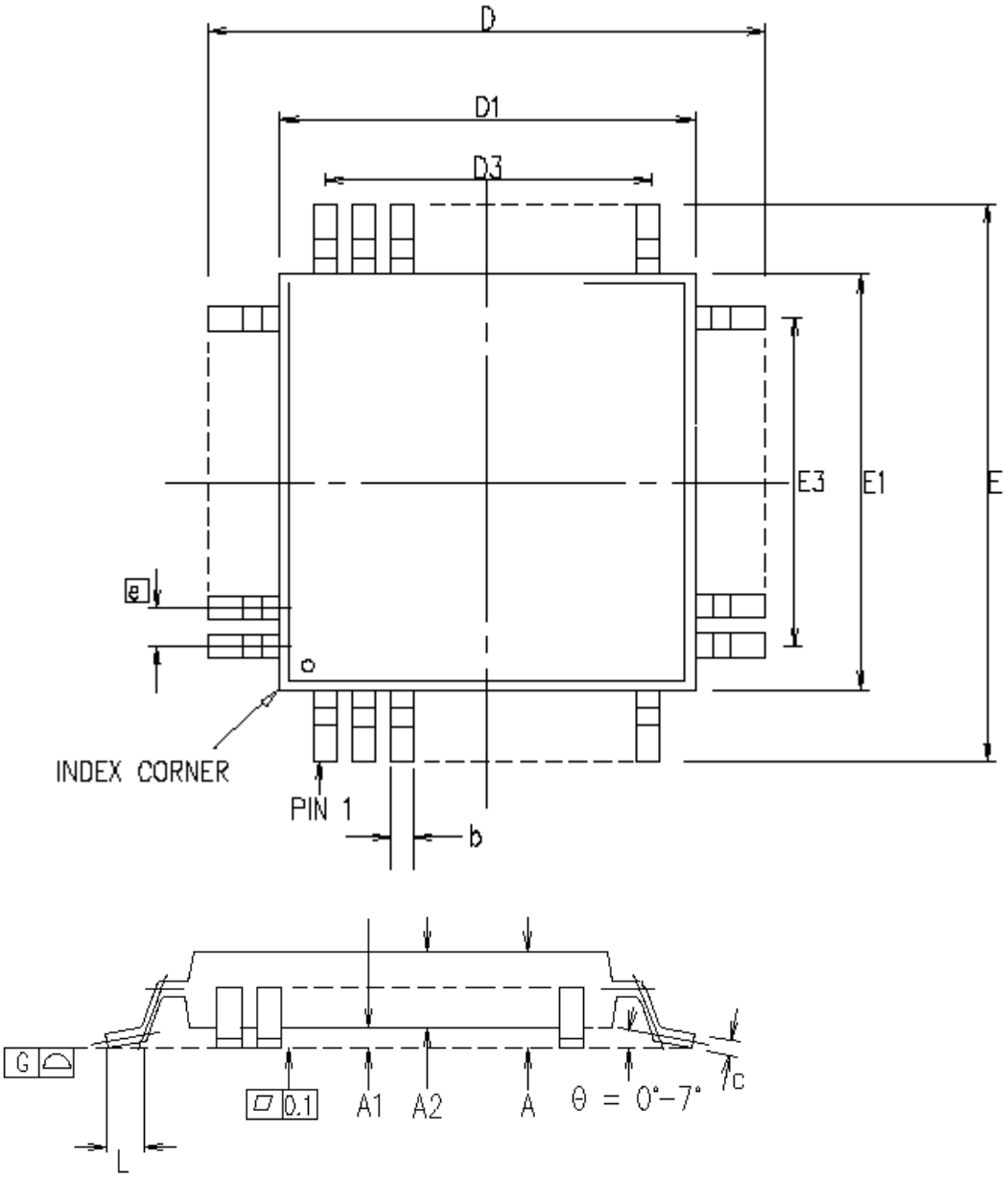


Figure 2.2 GP4020 100-pin package outline drawing

Symbol	Dimensions in millimetres		
	MIN	Nominal	MAX
A	1.40		1.60
A1	0.05		0.15
A2	1.35		1.45
D	15.80		16.20
D1	13.80		14.20
D3		12.00	
E	15.80		16.20
E1	13.80		14.20
E3		12.00	
L	0.45		0.75
e		0.50	
b	0.17		0.27
c	0.09		0.20

Table 2.1 GP4020 100-pin package dimensions

2.2 GP4020 100-pin Package Electrical Connection Details

All Vdd and GND pins must be connected to ensure reliable operation. Any unused input pins must be tied either High or Low; no inputs should be left unconnected.

Pin No.	Signal Name	Type	Circuit Block	Description	Notes
1	SADD[0]	I/O	MPC	System Address bit 0	
2	SADD[1]	I/O	MPC	System Address bit 1	
3	SADD[2]	I/O	MPC	System Address bit 2	
4	SADD[3]	I/O	MPC	System Address bit 3	
5	SADD[4]	I/O	MPC	System Address bit 4	
6	SADD[5]	I/O	MPC	System Address bit 5	
7	GND	PWR			
8	SADD[6]	I/O	MPC	System Address bit 6	
9	SADD[7]	I/O	MPC	System Address bit 7	
10	VDD	PWR			
11	NSCS[0]	I/O	MPC	System Chip Select 0 - Active Low	1
12	NSCS[1]	O	MPC	System Chip Select 1 - Active Low	1
13	NSCS[2A]	O	MPC	System Chip Select 2A - Active Low	1
14	SADD[19]	O	MPC	System Address bit 19	
15	SDATA[0]	I/O	MPC	System Data bit 0	1
16	SDATA[1]	I/O	MPC	System Data bit 1	1
17	SDATA[2]	I/O	MPC	System Data bit 2	1
18	SDATA[3]	I/O	MPC	System Data bit 3	1
19	GND	PWR			
20	SDATA[4]	I/O	MPC	System Data bit 4	1
21	SDATA[5]	I/O	MPC	System Data bit 5	1
22	VDD	PWR			
23	SDATA[6]	I/O	MPC	System Data bit 6	1

Pin No.	Signal Name	Type	Circuit Block	Description	Notes
24	SDATA[7]	I/O	MPC	System Data bit 7	1
25	NSOE	I/O	MPC	System Output Enable - Active Low	1
26	NSWE[1]	I/O	MPC	System Write Enable bit 1 - Active Low	1
27	NSWE[0]	I/O	MPC	System Write Enable bit 0 - Active Low	1
28	SDATA[8]	I/O	MPC	System Data bit 8	1
29	SDATA[9]	I/O	MPC	System Data bit 9	1
30	VDD	PWR			
31	SDATA[10]	I/O	MPC	System Data bit 10	1
32	SDATA[11]	I/O	MPC	System Data bit 11	1
33	GND	PWR			
34	SDATA[12]	I/O	MPC	System Data bit 12	1
35	SDATA[13]	I/O	MPC	System Data bit 13	1
36	SDATA[14]	I/O	MPC	System Data bit 14	1
37	SDATA[15]	I/O	MPC	System Data bit 15	1
38	SADD[18]	I/O	MPC	System Address bit 18	
39	SADD[17]	I/O	MPC	System Address bit 17	
40	SADD[16]	I/O	MPC	System Address bit 16	
41	GND	PWR			
42	SADD[15]	I/O	MPC	System Address bit 15	
43	SADD[14]	I/O	MPC	System Address bit 14	
44	VDD	PWR			
45	SADD[13]	I/O	MPC	System Address bit 13	
46	SADD[12]	I/O	MPC	System Address bit 12	
47	SADD[11]	I/O	MPC	System Address bit 11	
48	SADD[10]	I/O	MPC	System Address bit 10	
49	SADD[9]	I/O	MPC	System Address bit 9	
50	SADD[8]	I/O	MPC	System Address bit 8	
51	SWAIT	I	MPC	System Wait input - allows wait-states to be inserted into the current Firefly clock cycle.	
52	NSUB	O	MPC	System Upper Byte - Active Low	1,2
53	IEXTINT2	I	INTC	Interrupt source 2 input (for external interrupts)	
54	MULTI_FNIO	I/O	PCL	Multi-function Input / Output. Used to set Boot Up ROM area, and source either 100kHz square wave or System Clock	
55	DISCIO	I/O	PCL	Discrete Input / Output Used either as input or to source RF_Power_Down control signal or TIC.	3
56	RF_PLL_LOCK	I	INTC / PCL	PLL Lock Indicator input from RF section. When High this signal indicates that the PLL within the RF section is in lock and the master-clock inputs have stabilised.	
57	A1VDD	PWR		VDD Supply for CLK_T & CLK_I input block in the System Clock Generator. This pin should be well de-coupled to pin 60 (GND) to ensure optimum noise immunity.	
58	CLK_T	I	SCG	Master Clock (M_CLK) Input from RF Front-end - 40MHz 100mV rms.	4
59	CLK_I	I	SCG	Inverted Master Clock (M_CLK) Input from RF Front-end - 40MHz 100mV rms.	4
60	GND	PWR			

Pin No.	Signal Name	Type	Circuit Block	Description	Notes
61	SIGN0	I	CORR	Sampled Sign (polarity) data from RF Front-end	
62	MAG0	I	CORR	Sampled Mag (amplitude) data from RF Front-end	
63	SAMPCLK	O	CORR	Sample Clock output to the RF front end. Provides a 5.714MHz clock with a 4:3 mark–to–space ratio.	
64	POWER_GOOD	I	PCL	Power Monitor input. High for normal operation. Low forces the GP4020 into Power Down mode.	
65	PR_XOUT	O	SCG	System Clock Oscillator - crystal oscillator output for 10 to 16MHz crystal.	
66	PR_XIN	I	SCG	System Clock Oscillator - crystal oscillator input for 10 to 16MHz crystal.	
67	TEST	I		Test Select Pin. Used with TESTMODE (Pin 74). *This pin is reserved for TEST purposes only and should be connected to GND in normal operation.*	5
68	VDD	PWR			
69	TIMEMARK / TIC	O	1PPS	Timemark output. This pin can be used to produce a UTC-aligned 1PPS output, or TIC output	
70	IDDQTEST	I		*This pin is reserved for TEST purposes only and should be connected to GND in normal operation*	
71	GND	PWR			
72	RTC_XIN	I	RTC	Real-time Clock Crystal Oscillator input for 32kHz crystal.	
73	RTC_XOUT	O	RTC	Real-time Clock Crystal Oscillator output for 32kHz crystal	
74	TESTMODE	I		Test-mode Select Pin. Used with TEST (Pin 67). *This pin is reserved for TEST purposes only and should be connected to GND in normal operation.*	5
75	NSRESET	I	PCL	System Reset input	
76	U2TXD	O	UART2	UART 2 Transmit data output.	
77	U2RXD	I	UART2	UART 2 Receive data input.	3
78	U1TXD	O	UART1	UART 1 Transmit data output.	
79	U1RXD	I	UART1	UART 1 Receive data input.	3
80	PLLGND	PWR	SCG PLL	GND connection for PLL Block	
81	PLLVDD	PWR	SCG PLL	VDD connection for PLL Block	
82	GND	PWR			
83	PLLAT1	O	SCG PLL	System Clock Generator PLL Analog Test IO. *This pin is reserved for TEST purposes only and should be NOT connected in normal operation*	
84	NICE	I	JTAG / SSM MUX	ARM7TDMI operating mode and JTAG / SSM signal multiplex (pins 86, 87, 88, 89).	6
85	VDD	PWR			
86	TCK / bdiag[0] / XReq	I/O	JTAG / SSM	JTAG Test Clock / SSM Diagnostic broadcast output bdiag[0] / System Test control input XReq	6
87	TDI / bdiag[1] / XWrite	I/O	JTAG / SSM	JTAG Test Data In / SSM Diagnostic broadcast output bdiag[1] / System Test control input XWrite	6

Pin No.	Signal Name	Type	Circuit Block	Description	Notes
88	TDO / bdiag[2] / XBurst	I/O	JTAG / SSM	JTAG Test Data Out / SSM Diagnostic broadcast output bdiag[2] / System Test control input XBurst	6
89	TMS / bdiag[3] / XCon	I/O	JTAG / SSM	JTAG Test Mode Select / SSM Diagnostic broadcast output bdiag[3] / System Test control input XCon	6
90	NTRST	I	JTAG / SSM	JTAG Interface Reset or SSM debug interface multiplex (pins 86, 87, 88, 89).	6
91	GPIO[7] / PLLDT1	I/O	GPIO / SCG PLL	General Purpose Input / Output pin 7. Can be multiplexed to SCG PLL Digital Test Output (PLLDT1).	3
92	GPIO[6]	I/O	GPIO	General Purpose Input / Output pin 6.	3
93	GPIO[5] / DISCOP	I/O	GPIO / CORR	General Purpose Input / Output pin 5. Can be multiplexed to DISCOP discrete output from correlator core.	3
94	GND	PWR			
95	GPIO[4] / DISCIP1	I/O	GPIO / CORR	General Purpose Input / Output pin 4. Also directly connects to DISCIP1 on the 12-channel correlator.	3
96	GPIO[3] / BSIO_SS[1]	I/O	GPIO / BSIO	General Purpose Input / Output pin 3. Can be multiplexed to BSIO Slave Select[1].	3
97	GPIO[2] / BSIO_SS[0]	I/O	GPIO / BSIO	General Purpose Input / Output pin 2. Can be multiplexed to BSIO Slave Select[0].	3
98	VDD	PWR			
99	GPIO[1] / BSIO_DATA	I/O	GPIO / BSIO	General Purpose Input / Output pin 1. Can be multiplexed to BSIO Data Input / Output.	3
100	GPIO[0] / BSIO_CLK	I/O	GPIO / BSIO	General Purpose Input / Output pin 0. Can be multiplexed to BSIO_CLK output.	3

Table 2.2 GP4020 100-pin package Signal Descriptions

Notes:

1) Hi Impedance is achieved on pins 11 to 18, 20, 21, 23 to 29, 31, 32, 34 to 37 when either:
   (a) Data is not being written from GP4020
   (b) POWER_GOOD (pin 64) is Low;
   (c) Bit 1 ("RF_PD") of POW_CNTL register is high;
   (d) Bit 10 ("RF_SLEEP") of POW_CNTL register is High;

2) NSUB (pin 52 is the Upper Byte select output from the Memory Peripheral Controller, when single-chip 16-bit memories with NUB and NLB inputs are used. NSUB maps to NUB and address line SADD[0] to NLB.

3) Input is tolerant to being driven with a +5V HIGH level, as well as +3.3V HIGH nominal level.

4) Both CLK_T (pin 58) and CLK_I (pin 59) should not have an external DC bias of GREATER than +1.7V. Direct connection from a GP2010 / GP2015 RF Front-end is NOT possible, without a bias-shift circuit (refer to *Block Diagram of typical GP4020 based GPS receiver" on page 8*, and Section *14.2 "40MHz Low Level Differential Input" on page 136* for more information).

5) TEST (pin 67) and TESTMODE (pin 74) are used together to set-up 3 manufacturing test-modes for the GP4020:

TEST (pin 67)	TESTMODE (pin 74)	TEST FUNCTION
GND (0)	GND (0)	Normal Operation
VDD (1)	GND (0)	Firefly Macrocell test mode
GND (0)	VDD (1)	Firefly System test mode
VDD (1)	VDD (1)	UIM Logic test mode

Details of ALL test modes are covered in *section 2.10 of the Firefly MF1 Core Design Manual (DM5003)*, available from Zarlink Semiconductor.

6) NICE (pin 84) and NTRST (pin 90) control a number of operation modes and a debug signal multiplex on pins 86, 87, 88, 89 and 90, as follows:

NICE = Low ARM7TDMI in ICE mode.

ARM7TDMI will not access memory unless instructed to by the JTAG interface. NTRST (pin 90) set Low will reset the JTAG Interface.

NICE = High ARM7TDMI in Normal mode.

NTRST does not effect a reset on the JTAG interface. However, a reset of Firefly will also reset the JTAG.

NTRST (pin 90) has a reset and signal-multiplex function, dependent on the state of the NICE input (pin 84):

i) NICE = Low: JTAG debug signals connected to pins 86, 87, 88, 89 & 90, as follows:

Pin 86 = TCK = JTAG clock in

Pin 87 = TDI = JTAG data in

Pin 88 = TDO = JTAG data out

Pin 89 = TMS = JTAG mode select in

Pin 90 = NTRST = Active low reset to JTAG interface
(JTAG interface also reset when Firefly MF1 is reset)

ii) NICE = High and NTRST = High:

This is the Normal mode of operation for GP4020. The System Services Module Broadcast Diagnostic debug output signals are connected to pins 86, 87, 88, 89 as follows:

Pin 86 = BDIAG[0]

Pin 87 = BDIAG[1]

Pin 88 = BDIAG[2]

Pin 89 = BDIAG[3]

Diagnostic mode must have been set-up using the Diagnostic Configuration Registers within Firefly MF1. Refer to *Section 8 of Firefly MF1 Core Design Manual (DM5003)*, from Zarlink Semiconductor, for more information.

iii) NICE = High and NTRST = Low:

Firefly MF1 System Test Control input signals are connected to pins 86, 87, 88, and 89 as follows:

Pin 86 = Xreq

Pin 87 = XWrite

Pin 88 = Xburst

Pin 89 = XCon

System test inputs are used in Firefly MF1 macrocell test mode for manufacturing test. Refer to *Section 2.10 of Firefly MF1 Core Design Manual (DM5003),* from Zarlink Semiconductor, for more information.

3 ARM7TDMI MICROPROCESSOR

The ARM7TDMI is a member of the Advanced RISC Machines (ARM) family of general-purpose 32-bit microprocessors, which offer high performance for very low power consumption and price. The ARM architecture is based on Reduced Instruction Set Computer (RISC) principles, and the instruction set and related decode mechanism are much simpler than those of micro-programmed Complex Instruction Set Computers. This simplicity results in a high instruction throughput and impressive real-time interrupt response from a small and cost-effective core.

Pipelining is employed so that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory.

The ARM memory interface has been designed to allow the performance potential to be realised without incurring high costs in the memory system. Speed-critical control signals are pipelined to allow system control functions to be implemented in standard low-power logic, and these control signals facilitate the exploitation of the fast local access modes offered by industry standard dynamic Ram.

The ARM7TDMI microprocessor is surrounded by a scan chain. This allows it to be isolated from the embedded system for debug purposes. Register or memory values may be examined, and breakpoints and watchpoints may be set via the JTAG interface. As ARM instructions are conditionally executed, the microprocessor pipeline follows breakpoints to determine whether a trigger condition exists.

3.1 ARM7TDMI Instruction Set Architecture

The ARM7TDMI microprocessor employs a unique architectural strategy known as Thumb, which makes it ideally suited to high-volume applications with memory restrictions or applications where high code density is essential.

The ARM 32-bit instruction set offers flexibility in instruction format and operand manipulation while producing maximum performance from 32-bit memory systems.

3.2 The Thumb Concept

The essential idea behind Thumb is that of a super-reduced instruction set. Essentially, the ARM7TDMI microprocessor has two instruction sets. The Thumb set’s 16-bit instruction length allows it to approach twice the density of standard ARM code while retaining most of the ARM7TDMI's performance advantage over a traditional 16-bit microprocessor using 16-bit registers. This is possible because Thumb code operates on the same 32-bit register set as ARM code.

Thumb code is able to provide up to 65% of the code size of ARM, and 160% of the performance of an equivalent ARM microprocessor connected to a 16-bit memory system.

3.3 Thumb’s Advantages

Thumb instructions operate with the standard ARM register configuration, allowing excellent interoperability between ARM and Thumb states. Each 16-bit Thumb instruction has a corresponding 32-bit ARM instruction with the same effect on the microprocessor model.

The major advantage of a 32-bit (ARM) architecture over a 16-bit architecture is its ability to manipulate 32-bit integers with single instructions, and to address a large address space efficiently. When processing 32-bit data, a 16-bit architecture will take at least two instructions to perform the same task as a single ARM instruction.

However, not all the code in a program will process 32-bit data (for example, code that performs character string handling), and some instructions, like Branches, do not process any data at all.

If a 16-bit architecture only has 16-bit instructions, and a 32-bit architecture only has 32-bit instructions, then overall the 16-bit architecture will have better code density. Also 16-bit will have better than one half the performance of the 32-bit architecture.

Clearly 32-bit performance comes at the cost of code density. Thumb breaks this constraint by implementing a 16-bit instruction length on a 32-bit architecture, making the processing of 32-bit data efficient with a compact instruction coding. This provides far better performance than a 16-bit architecture, with better code density than a 32-bit architecture.

Thumb also has a major advantage over other 32-bit architectures with 16-bit instructions. This is the ability to switch back to full ARM code and execute at full speed. Thus critical loops for applications such as fast interrupts, DSP algorithms can be coded using the full ARM instruction set, and linked with Thumb code. The overhead of switching from Thumb code to ARM code is folded into sub-routine entry time. Various portions of a system can be optimised for speed or for code density by switching between Thumb and ARM execution as appropriate.

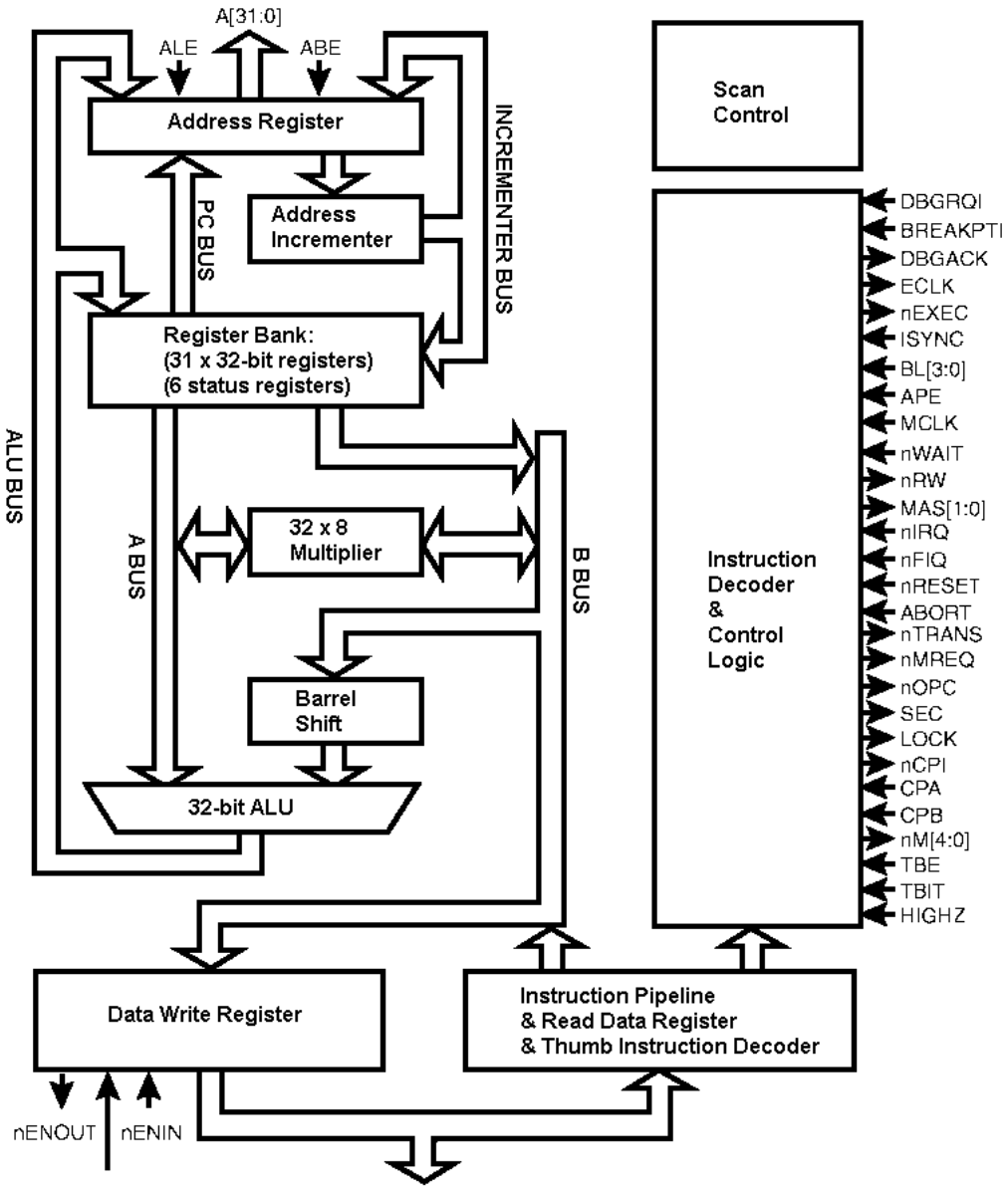


Figure 3.1 ARM7TDMI Architecture

Mnemonic	Instruction	Action
ADC	Add with carry	Rd := Rn + Op2 + Carry
ADD	Add	Rd := Rn + Op2
AND	AND	Rd := Rn AND Op2
B	Branch	R15 := address
BIC	Bit Clear	Rd := Rn AND NOT Op2
BL	Branch with Link	R14 := R15, R15 := address
BX	Branch and Exchange	R15 := Rn, T bit := Rn[0]
CDP	Coprocessor Data Processing	(Coprocessor-specific)
CMN	Compare Negative	CPSR flags := Rn + Op2
CMP	Compare	CPSR flags := Rn - Op2
EOR	Exclusive OR	Rd := (Rn AND NOT Op2) OR (op2 AND NOT Rn)
LDC	Load coprocessor from memory	Coprocessor load
LDM	Load multiple registers	Stack manipulation (Pop)
LDR	Load register from memory	Rd := (address)
MCR	Move CPU register to coprocessor register	cRn := rRn {<op>cRm}
MLA	Multiply Accumulate	Rd := (Rm * Rs) + Rn
MOV	Move register or constant	Rd : = Op2
MRC	Move from coprocessor register to CPU register	Rn := cRn {<op>cRm}
MRS	Move PSR status/flags to register	Rn := PSR
MSR	Move register to PSR status/flags	PSR := Rm
MUL	Multiply	Rd := Rm * Rs
MVN	Move negative register	Rd := 0xFFFF FFFF EOR Op2
ORR	OR	Rd := Rn OR Op2
RSB	Reverse Subtract	Rd := Op2 - Rn
RSC	Reverse Subtract with Carry	Rd := Op2 - Rn - 1 + Carry
SBC	Subtract with Carry	Rd := Rn - Op2 - 1 + Carry
STC	Store coprocessor register to memory	address := CRn
STM	Store Multiple	Stack manipulation (Push)
STR	Store register to memory	<address> := Rd
SUB	Subtract	Rd := Rn - Op2
SWI	Software Interrupt	OS call
SWP	Swap register with memory	Rd := [Rn], [Rn] := Rm
TEQ	Test bitwise equality	CPSR flags := Rn EOR Op2
TST	Test bits	CPSR flags := Rn AND Op2

Table 3.1 Standard 32-bit ARM instruction set

Mnemonic	Instruction	Action	Lo/Hi register operands	Condition codes set
ADC	Add with Carry	Rd := Rd + Rs + C	Lo	Yes
ADD	Add	Rd := Rn + Rs	Lo/Hi	Yes*
AND	AND	Rd := Rd AND Rs	Lo	Yes
ASR	Arithmetic Shift Right	Rd := Rd ASR Rs	Lo	Yes
B	Unconditional branch	PC := PC +/- Offset11	Lo	
Bxx	Conditional branch	PC := PC +/- Offset8	Lo	
BIC	Bit Clear	Rd := Rd AND NOT Rs	Lo	Yes
BL	Branch and Link	PC := PC +/- Offset	LR:=PC + 2	
BX	Branch and Exchange	PC := Rs	Lo / Hi	
CMN	Compare Negative	Rd + Rs	Lo	Yes
CMP	Compare	CPSR flags :=Rd - Rs	Lo / Hi	Yes
EOR	EOR	Rd := Rd EOR Rs	Lo	Yes
LDMIA	Load multiple	Stack manipulation (Pop)	Lo	
LDR	Load word	Rd32 := [Rb + Immediate5]	Lo	
LDRB	Load byte	Rd8 := [Rb + Immediate5]	Lo	
LDRH	Load half-word	Rd16 := [Rb + Immediate5]	Lo	
LSL	Logical Shift Left	Rd := Rd << Rs	Lo	Yes
LDSB	Load sign-extended byte	Rd8 := [Rb + Immediate5]	Lo	
LDSH	Load sign-extended half-word	Rd16 := [Rb + Immediate5]	Lo	
LSR	Logical Shift Right	Rd := Rd >> Rs	Lo	Yes
MOV	Move register	Rd := Immediate8	Lo / Hi	Yes*
MUL	Multiply	Rd := Rs * Rd	Lo	Yes
MVN	Move Negative register	Rd := NOT Rs	Lo	Yes
NEG	Negate	Rd := -Rs	Lo	Yes
ORR	OR	Rd := Rd OR Rs	Lo	Yes
POP	Pop registers	[SP] ++ := Rlist (LR)	Lo	
PUSH	Push registers	Rlist (LR):= [SP] --	Lo	
ROR	Rotate Right	Rd := Rd ROR Rs	Lo	Yes
SBC	Subtract with Carry	Rd := Rd -Rs - NOT C	Lo	Yes
STMIA	Store Multiple	[Rb]++ := Rlist	Lo	
STR	Store word	[Rb + Immediate5] := Rd32	Lo	
STRB	Store byte	[Rb + Immediate5] := Rd8	Lo	
STRH	Store half-word	[Rb + Immediate5] := Rd16	Lo	
SWI	Software Interrupt	OS call		
SUB	Subtract	Rd := Rd - Immediate8	Lo	Yes
TST	Test bits	CPSR flags :=Rd AND Rs	Lo	Yes

Table 3.2 16-bit Thumb instruction set

3.4 Operating Modes

ARM7TDMI supports seven modes of operation:

- User (usr) The normal ARM program execution state
- FIQ (fiq) Designed to support a data transfer or channel process
- IRQ (irq) Used for general-purpose interrupt handling
- Supervisor (svc) Protected mode for the operating system
- Abort mode (abt) Entered after a data or instruction pre-fetch abort

- System (sys) A privileged user mode for the operating system
- Undefined (und) Entered when an undefined instruction is executed

Mode changes may be made under software control, or may be brought about by external interrupts or exception processing. Most application programs will execute in User mode. The non-user modes ("privileged modes") are entered in order to service interrupts or exceptions, or to access protected resources.

3.5 Register Sets

In ARM State, 16 general registers and 1 or 2 status registers are visible at any one time. In privileged (non-User) modes, mode-specific banked registers are switched in. *Table 3.3, Table 3.4, Table 3.5, and Table 3.6 below* show which registers are available in each mode: an asterisk indicates the banked registers (*):

System & User	FIQ	Supervisor	Abort	IRQ	Undefined
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
R8	R8_fiq *	R8	R8	R8	R8
R9	R9_fiq *	R9	R9	R9	R9
R10	R10_fiq *	R10	R10	R10	R10
R11	R11_fiq *	R11	R11	R11	R11
R12	R12_fiq *	R12	R12	R12	R12
R13	R13_fiq *	R13_svc *	R13_abt *	R13_irq *	R13_und *
R14	R14_fiq *	R14_svc *	R14_abt *	R14_irq *	R14_und *
R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)

Table 3.3 ARM State General Registers and Program Counter

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_fiq *	SPSR_svc *	SPSR_abt *	SPSR_irq *	SPSR_und *

Table 3.4 ARM State Program Status Registers

* Indicates register is banked.

The Thumb State register set is a subset of the ARM State set. The programmer has direct access to eight general registers, R0-R7, as well as the Program Counter (PC), a stack pointer register (SP), a link register (LR), and the CPSR. There are banked Stack Pointers, Link Registers and Saved Process Status Registers (SPSRs) for each privileged mode.

System & User	FIQ	Supervisor	Abort	IRQ	Undefined
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
SP	SP_fiq *	SP_svc*	SP_abt *	SP_irq *	SP_und *
LR	LR_fiq *	LR_svc *	LR_abt *	LR_irq *	LR_und *
PC	PC	PC	PC	PC	PC

Table 3.5 Thumb State General Registers and Program Counter

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_fiq *	SPSR_svc *	SPSR_abt *	SPSR_irq *	SPSR_und *

Table 3.6 Thumb State Program Status Registers

* Indicates register is banked.

3.6 Low Power ARM7TDMI Sleep Mode

A feature of the Firefly MF1 ARM7TDMI, is a Sleep Coprocessor, which can be used to disable the clock to the ARM7TDMI, but keep it enabled to other parts of the Firefly MF1. This is different to the F_SLEEP utility in the Peripheral Control Logic block of the GP4020, as it allows all other Firefly blocks to remain operable while the ARM7TDMI is halted.

The ARM7TDMI core does not inherently contain a low power sleep mode; however, the architecture does contain a mechanism for instruction set extension through the coprocessor interface. Zarlink has taken advantage of this interface to define a coprocessor instruction set implementing a low power operation (sleep) mode.

This coprocessor is assigned coprocessor number "3", and performs Coprocessor Data operations (CDP). In the implementation contained within the Firefly MF1 micro-controller, one coprocessor instruction (“0”) is defined:

0 Suspend processor operation and halt processor clock until interrupt is received from any enabled interrupt in the INTC block. Upon receipt of interrupt, execute an interrupt service routine, then resume the normal flow of execution after the CDP instruction.

All other instructions for that coprocessor and all other coprocessor instruction types are reserved. The assembly code for the SLEEP instruction is:

```
NOP
NOP
CDP p3,0,c0,c0,c0

; Firefly Arm7 Sleep enable, Coprocessor no. 3, Operation ‘0’

; The ARM CPU has now been halted. Other Masters may still operate.

; An interrupt to the ARM will reawaken the ARM.

NOP
NOP
```

An interrupt impulse to the ARM7TDMI will cause it to exit SLEEP mode. In certain circumstances, this may cause the ARM7TDMI to enter an UNDEF (Undefined Instruction) trap (to address 0x04). In order to return to normal program control, a:

```
MOVS PC,R14_und
```

instruction should be placed at address 0x04. If the UNDEF trap is to be used for other purposes also, a test starting at location 0x04 will be necessary, to identify if the trap was as a result of an interrupt whilst in SLEEP mode.

Note: With the ARM7TDMI processor in SLEEP mode, other Bus Masters (e.g.: DMAC) are still able to utilise the bus.

3.6.1 Info on the Undefined Instruction Trap

When using the Sleep co-processor, the ARM7TDMI can get an instruction when re-enabled, which it cannot handle, and it will take an “Undefined Instruction” trap. The trap may be used for software emulation of a coprocessor in a system, which does not have the coprocessor hardware, or for general purpose instruction set extension by software emulation.

When ARM7TDMI takes the undefined instruction trap, it performs the following:

1) Saves the address of the Undefined or coprocessor instruction plus four in “R14_und”; saves CPSR in “SPSR_und”.

2) Forces M[4:0]=’11011’ (Undefined mode) and sets the I bit in the CPSR

3) Forces the PC to fetch the next instruction from address 0x04

To return from this trap after emulating the failed instruction, use:

```
MOVS PC,R14_und.
```

This will restore the CPSR and return to the instruction following the undefined instruction.

Further details of the function and programming of the ARM7TDMI microprocessor can be found in *Section 2 of the "Firefly MF1 Core Design Manual" DM5003,* available from Zarlink Semiconductor. In addition Rev 3 of the ARM7TDMI Technical Reference Manual (document reference ARM DDI 0029F), is downloadable (1.7 MB PDF) from ARM's website http://www.arm.com. The documentation download page can be found at: http://www.arm.com/arm/documentation?OpenDocument.

This page intentionally left blank.

4 BOOT ROM

4.1 Functional Description

The GP4020 Boot ROM is an internal part of the IC. The code in the Boot ROM will allow the GP4020 based GPS receiver to up-load a software routine into RAM from an external data source (e.g. a PC), and run the routine from RAM. The uploaded routine could be used to update the GPS application firmware stored in FLASH EPROM. The Boot ROM does NOT need to run every time the GP4020 is powered up. There are a number of methods used to select either the internal Boot ROM or an External ROM.

The Boot ROM contains code that executes from address 0x0000 0000 via Firefly address area select line NCS[0]. The GP4020 can be configured to use either the internal ROM for system boot-up or an external ROM. This can be influenced by whether the GP4020 based GPS receiver is:

a) running final application software which can boot from an external ROM;

b) running a software utility is downloaded via UART1, booted using the internal ROM.

The GP4020 contains two mechanisms to select whether the NCS[0] signal from the Firefly MPC addresses the internal boot-ROM, or an external device connected to NSCS[0] (pin 11 (100-pin package)):

1) Control bit EXT_NCS0 in the IO_REV register within the PCL block.

2) A Latch that stores the state of MULTI_FNIO (pin 54 (100-pin package)) at the end of a chip reset due to the NPOR_RESET signal. Refer to *Section 12.2 "Chip Reset Logic" on page 113* for details on this reset mechanism.

Setting EXT_NCS[0] (bit 9 of the IO_REV register) can disable the Boot ROM. However, if MULTI_FNIO is low during a reset of the Firefly MF1 core, this control bit has no effect; it cannot override the hardware disable.

The Boot ROM is a maximum size of 512 words (16-bits wide). The code execution is dependent upon the input value on the MULTI_FNIO pin. This pin is by default set to be an input, although this can be re-configured with application software by the Peripheral Control Logic, after the Boot Code has run.

A reset due to NSRESET (pin 75 (100-pin package)) going Low, or a Watchdog time-out, will cause the internal signal NPOR_RESET to be active (low).

NPOR_RESET will cause the EXT_NCS0 bit in the IO_REV register to be cleared.

In addition, at the rising edge of NPOR_RESET, the state of MULTI_FNIO is latched. The latched version of MULTI_FNIO is inverted, to generate a signal called INT_NCS[0].

If INT_NCS[0] is high (i.e. MULTI_FNIO (pin 54 (100-pin package)) was low at rising edge of NPOR_RESET) OR EXT_NCS0 is high, then NCS[0] from Firefly selects external NSCS[0] ROM device. Otherwise, the internal boot-ROM is selected.

A Read of the EXT_NCS0 bit in the PCL IO_REV register, will return the actual NCS[0] source selected (i.e. If INT_NCS[0] is high, EXT_NCS0 bit will always be read as '1', irrespective of what is written to it. If INT_NCS[0] has been used to select the external NSCS[0], the EXT_NCS0 bit cannot be used to switch back to using internal boot-ROM).

If the internal boot-ROM is selected, when the Firefly reset is released, it will:

1) Download data from UART1 and store it in internal RAM (using the protocol defined below);

2) Use an MPC function (configured in the SSM System Configuration Register) which swaps the address ranges of the Firefly Memory area Select lines, NCS[0] and NCS[3]. (i.e. the internal boot-ROM and NSCS[0], with the Internal RAM space). This effectively swaps the memory space of the internal SRAM (0x6000 0000), with the space for the Internal Boot ROM (0x0000 0000). The internal SRAM space then effectively starts from 0x0000

0000, and the ROM space from 0x6000 0000. The ARM7TDMI will then begin execution of code downloaded to the Internal RAM, starting at address 0x6000 0000.

The EXT_NCS0 bit in the IO_REV register (within the PCL) can then be set so that Firefly NCS[0] signal selects external NSCS[0] instead of internal boot-ROM. Remember that NSCS[0] will now start at address 0x6000 0000 due to MPC swap.

The Memory space swap implemented by the boot ROM can be cleared by a GP4020 reset, due to:

1) RF_PLL_LOCK (pin 56 (100-pin package)) going Low, with EN_PLL_RST set to '1' (bit 7 of PER_STAT register in PCL);

2) POWER_GOOD (pin 64 (100-pin package)) going Low, with EN_POW_RST set to '1' (bit 6 of PER_STAT register in PCL);

3) SFT_RESET set to a '1' (bit 4 of PER_STAT register in PCL);

This will cause the MPC swap function to revert to NCS[0] appearing at address 0x0000 0000, and NCS[3] appearing at address 0x6000 0000. However, these three reset sources will not effect whether NCS[0] selects internal boot-ROM or external NSCS[0] device.

4.2 UART Download Data Protocol

The nominal UART1 speed is set to be ~57.6Kbaud, and the UART clock is derived from the 40MHz CLK_T and CLK_I signals from the RF Front-end IC. The UART clock is 20MHz, and the UART1 Baud-Rate-Register is set to produce a 16 x Baud Rate of 0.90909MHz. The actual baud rate is 56.8kBaud, which is in error by -1.3%.

The protocol to be used for downloading data to the GP4020 is detailed below. The main purpose of the protocol is to provide simple but reliable data transfer. The protocol does not include any error checking. Any error checking on the downloaded code can be performed by the downloaded code itself when it starts to execute. This has two advantages:

1) It keeps the download protocol simple.

2) It allows maximum flexibility in the error checking routines that can be implemented.

The data protocol has three Header Bytes. These provide an indication of the number of Data Bytes (N) which are to be transmitted and hence stored in the internal RAM area of the GP4020 (this number excludes the three header bytes). The first header Byte (Byte 1) is the most significant data for a 24-bit number, Byte 2 is the next most significant and Byte 3 is the least significant.

Once the header Bytes have been transmitted, the data bytes can be sent. The boot ROM will cycle through a retrieve-and-store routine for each byte in the transmitted data, upto a total of N times. Each data byte will be stored in the Internal RAM. *Figure 4.1 below* shows the structure of the download data.

When the last byte has been transmitted, the internal ROM and internal RAM address areas are swapped and program execution will then start from address 0x0000 0000 in internal RAM. This is achieved by swapping Firefly Chip select lines NCS[0] and NCS[3], so effectively the internal Boot ROM will appear at address 0x6000 0000.

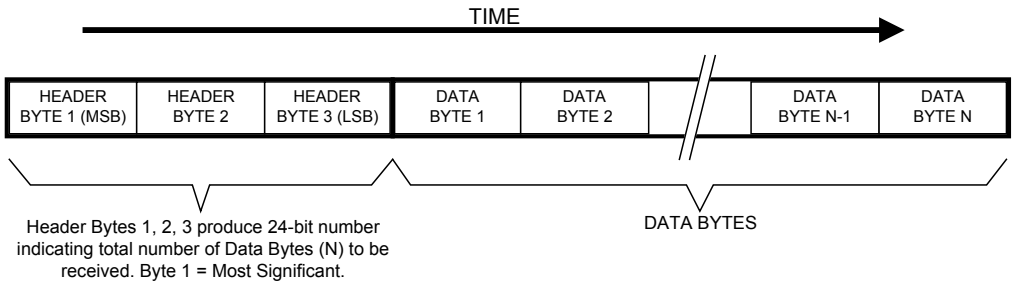


Figure 4.1 Boot ROM UART Download Data Protocol

This Page intentionally left blank.

5 The BμILD BUS

The GP4020 Baseband Processor CPU subsystem is internally based around the BμILD bus. The ARM7TDMI processor is connected to peripherals through its **B**us for μController **I**ntegration in **L**ow-Power **D**esigns (BμILD).

Although the GP4020 user does NOT need to know details of the internal operation of the BμILD bus for most applications, the implementation details are included for information.

This section contains a technical overview of the protocols associated with bus arbitration and bus transactions. This represents sufficient information to give a working knowledge of the implementation of the BμILD Bus within this embedded ARM system. The BμILD architecture is optimised for efficient on-chip embedded systems. It is primarily designed to support ARM CPUs and support modules, but is extensible to other processors and logic.

The following text describes the essential aspects of BμILD including the principal functional elements and protocol definitions.

5.1 Bus Masters

The bus master is the controller of the current bus transaction. A bus master initiates bus requests, generates addresses and controls data transfers while it has bus access, by reading or writing data over the data bus.

Bus masters on the GP4020 are:

- The ARM7TDMI CPU
- Direct Memory Access (DMA) multi-channel controller(s)
- System Services Module (SSM) for external test and debug

5.2 Bus Slaves

A bus slave responds to addresses present on the internal Bus that are in its allocated range within the address map. It supplies or receives data during read or write cycles on demand. A slave may set a wait signal to delay access using the synchronous bus transfer protocol.

Example slave devices are:

- UART
- Memory / Peripheral Controller
- General Purpose Input Output

5.3 Bus Signals

The BμILD bus, internal to the GP4020 has full 32-bit un-multiplexed address and data busses, b_addr<31:0> and b_data<31:0>. The direction of the current transaction is denoted by a write not read signal, b_write. The BμILD bus also supports multiple transaction sizes of byte, half-word (16-bits) and word (32-bits), as denoted by b_size<1:0>. Along with these main control signals are a number of additional control signals such as b_mode<2:0> that specifies the current bus-operating mode. In addition, two control signals are driven by the current bus slave, b_wait and b_error. b_wait is used to denote that wait states must be inserted in to the current bus access while b_error is used to denote that the current bus transaction is illegal e.g. a write to a read-only register.

6 BμILD SERIAL INPUT OUTPUT (BSIO) INTERFACE

6.1 Overview

A 3-wire serial input/output is included in the GP4020 to allow serial data connection to any device with a three-pin serial interface. The BSIO pins are multiplexed with the General Purpose Input Output (GPIO) pins within the Peripheral Control Logic block.

Two serial select pins allow for multiple types of devices to be connected using the same clock and data line. The block is sufficiently configurable to connect to a variety of devices.

The device is primarily designed to connect to external EEPROM, but can also be used with any device with a standard 3-wire serial interface. Eight registers control the serial bus.

6.1.1 Design Features

The main features of the BSIO module are:

- MICROWIRE™ Interface compatibility, to allow interfacing to memory and peripheral devices supporting this standard
- Serial Peripheral Interface (SPI™) compatibility; an interface found on some Motorola, TI and ST Microcontrollers
- Data transfer with either byte or word oriented protocols
- Triple-buffered transmit and receive channels
- Operation in either Interrupt or Polled mode
- Support for upto two slave devices

6.1.2 Pinout

Pin Name	Direction	Function
BSIO_CLK	Output	Serial Clock Output
BSIO_DATA	Input / Output	Serial Data
BSIO_SS[1:0]	Output	Serial Select

6.1.3 Architecture

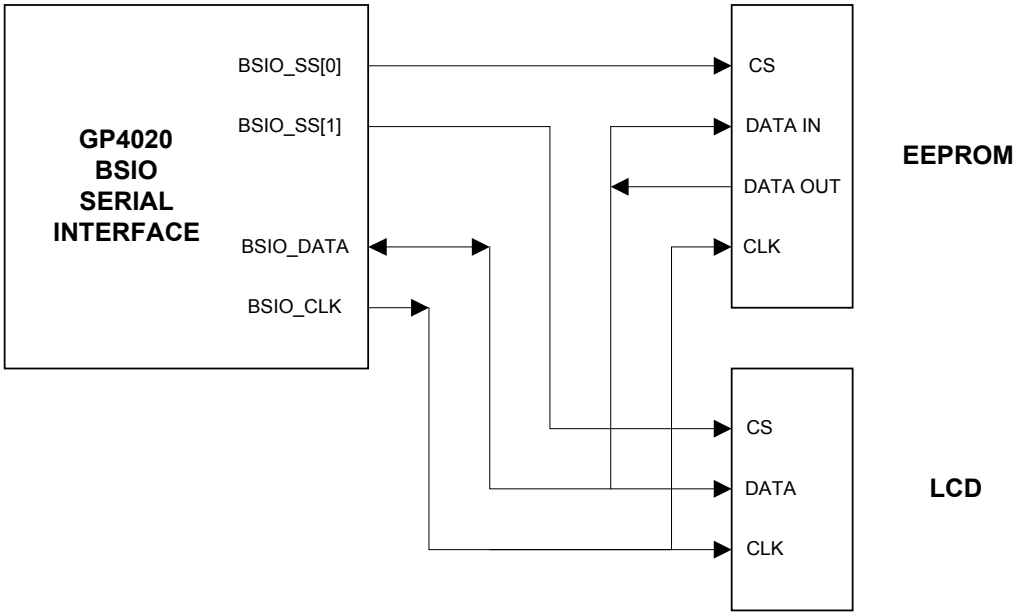


Figure 6.1 Using BµILD Serial Input Output (BSIO) with EEPROM and LCD peripherals

6.2 Operational Description

A control/status register configures the interface for each of the three select lines. A transfer register sets up individual transfers with the number of words to write and read. A data register allows incoming data to be read when in read mode and written when in write mode. An interrupt tells the ARM7TDMI when to read or write the data register. If interrupts are disabled, the status register may be read to poll for when to read/write the data register.

The transfer register is used to initiate all transfers over the serial bus. Each write to the transfer register starts a sequence of reads and writes over the bus directed by the data in the register. There are three possible scenarios for transfers; write; read; write then read. In the read scenario after the transfer word is written, the chosen chip select is asserted and data is read into the read buffer for the number of bytes required.

An interrupt is generated after each four bytes are read and at the end of the transfer to allow the ARM7TDMI to read out the new word of data. Write mode works similarly where the data is written over the serial bus, with an Interrupt occurring every four bytes of data. Write/Read mode starts with a number of bytes written over the interface followed by a number of bytes read over the interface. A control bit allows for a one-cycle delay between write and read for devices that require it. Write interrupts are generated during the write phase and then read interrupts are generated during the read phase.

Example of a write of five bytes:

- The ARM7TDMI writes the first four bytes to the RWBUF register.
- The ARM7TDMI writes the control information to the transfer register.
- The Serial block copies the RWBUF register to an internal register and generates a write interrupt to the ARM7TDMI to notify the RWBUF buffer is empty and begins sending data.
- The ARM7TDMI writes the last byte to the RWBUF register.
- The Serial block copies the RWBUF register to an internal register and generates a write interrupt to the ARM7TDMI to notify the RWBUF buffer is empty and begins sending data once the first four bytes have been sent.

- After all data has been sent, since there is no read data, a read data interrupt is generated immediately. If reading data, a read interrupt would be generated after each four bytes of data are read and after the last byte of data is read.

The BSIO consists of six blocks, the Sequencer, Frequency Divider, Write Buffer, Read Buffer, SIave Select Logic and Interrupt Control. A block diagram for the BSIO is shown in *Figure 6.2 below*.

Since the BSIO is an external Master, the only operations that are provided are a Read or a Write to a Slave. A Write operation consists of sending between zero to 1023 bytes/words to a Slave. A Read operation consists of first optionally sending between 0 to 1023 bytes/words to a Slave, and next receiving 0 to 1023 bytes/words.

The timing for a Read and Write operation from the BSIO to a Slave device is shown in *Figure 6.3 and Figure 6.4 below*. Refer to “Electrical Characteristics” in the “GP4020 GPS Baseband Processor Datasheet”, DS5134 for values of these timing parameters.

Note that the clock to the Slave devices SCLK, is static, i.e. held in either the High or Low state, whilst no operation is in progress.

Either byte or word transfers for write and read cycles are possible, with the word width being configurable between 2-bits and 32-bits. Data currently being transmitted / received is held in a 32-bit shift register. In addition, data that has been received or is awaiting transmission is held in a pair of 2 x 32-bit FIFOs. When byte transfers are used, the 32-bit word in the Read/Write buffers is treated as four consecutive bytes. In this case the lower order byte is transmitted first and is the first byte to be received. If however word transfers are used, then the selected number of lower order bits will make up the word, with any remaining higher order bits not being used. For example if a 20-bit Word were to be selected, then bdata<19:0> would make up the word, with bdata<31:20> not being used.

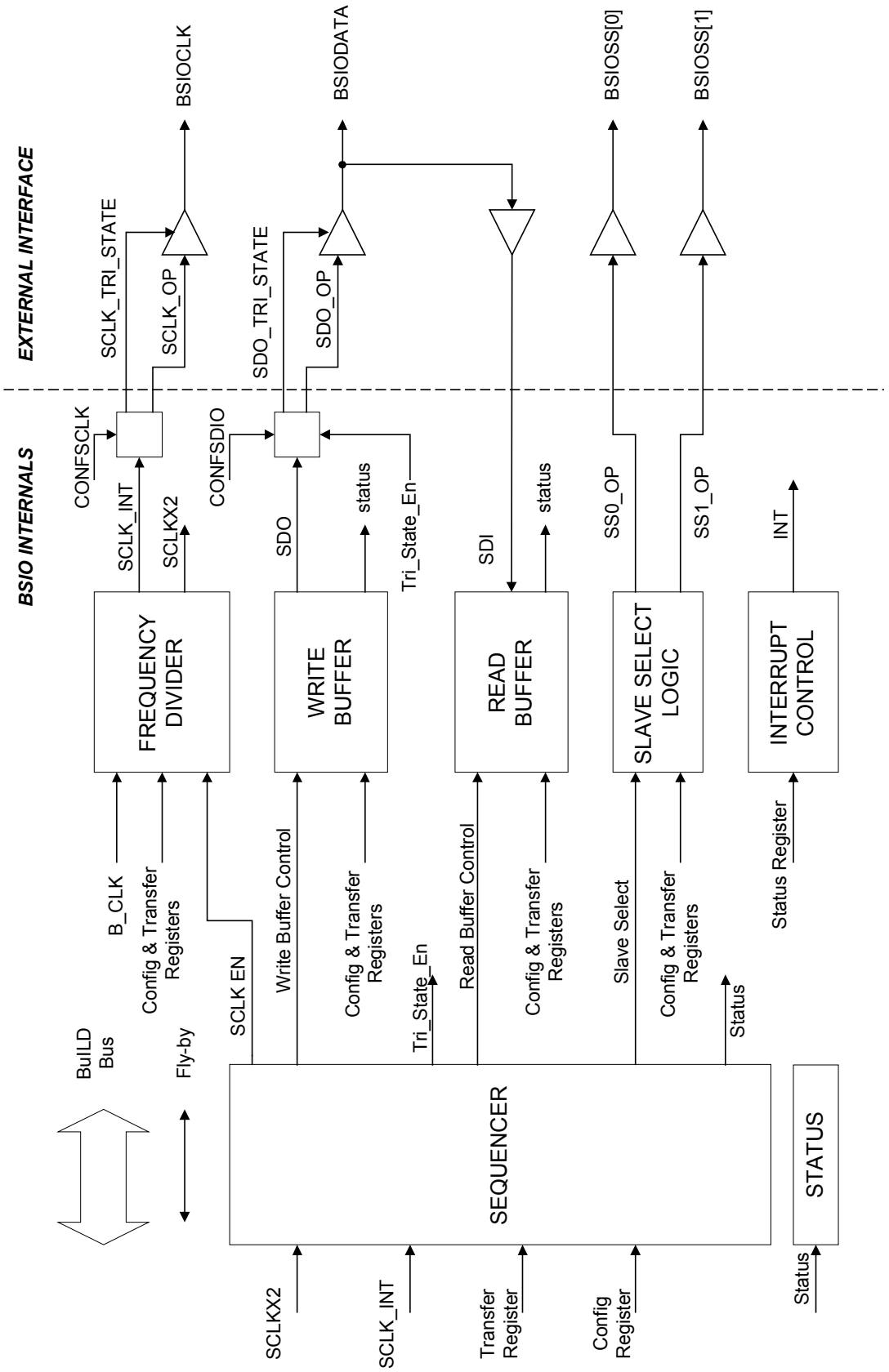


Figure 6.2 BµILD Serial Input Output (BSIO) Block Diagram

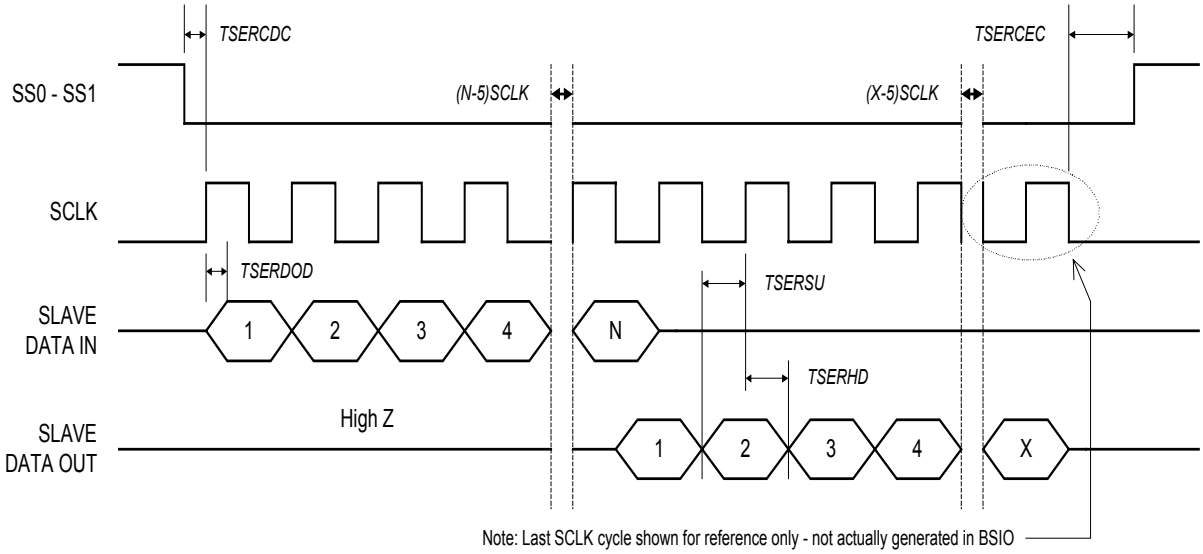


Figure 6.3 BSIO Read Operation Timing Diagram

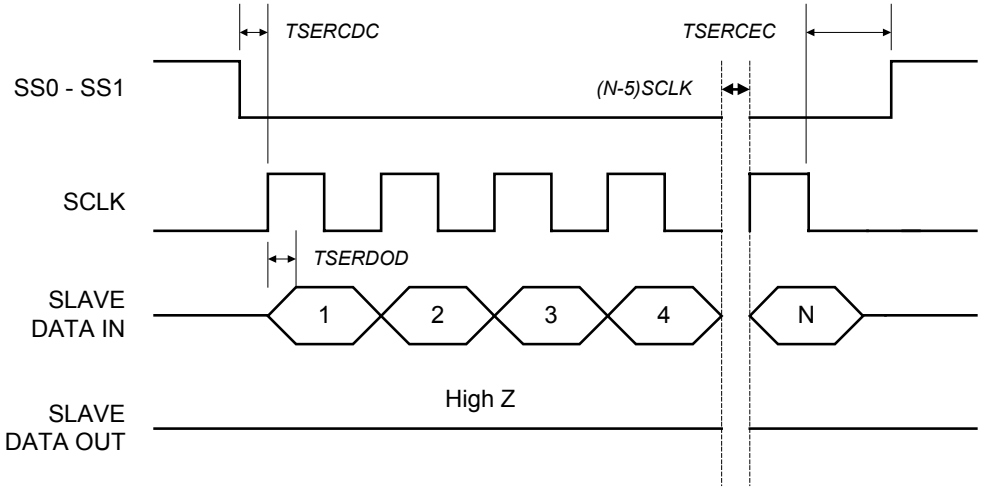


Figure 6.4 BSIO Write Operation Timing Diagram

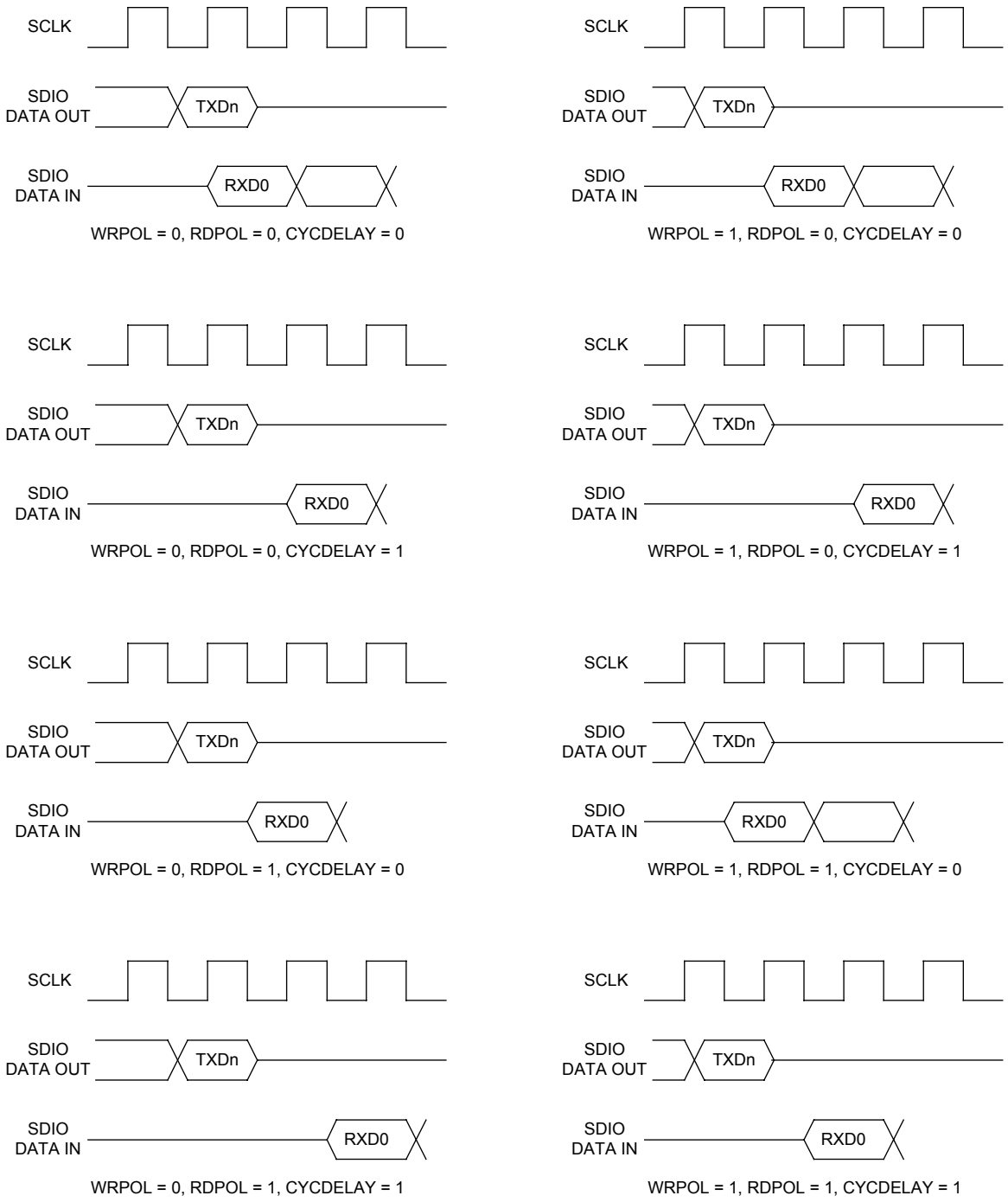


Figure 6.5 BSIO Bit timing options

In addition, the number of bytes/words to be sent and received in an operation is programmable between zero to 1023. Bit transfers will occur, on either the rising or falling edge, specified independently for read and write cycles via the RDPOL and WRPOL bits. This has the option of a 1-cycle delay between write and read cycles controlled by the CYCDELAY bit. The various bit timings possible are as shown in *Figure 6.5 above*.

In some applications, it may be necessary to send a Control Word at the start of an Operation. In order to support this the BSIO provides a Page Mode in addition to the Standard Mode described above. The Control Word is held in a separate 32-bit Parallel In / Serial Out Register. The CWORDSEL bit in the Mode Register selects between

Standard and Page Modes, with the width of the Control Word being configurable between 2-bits and 32-bits via the CWORD bits.

In Standard Mode, the start of an Operation is defined as when the first word is written to the Read/Write Buffer. In Page Mode, the start of an operation is defined as when the control word is written to the control word buffer.

In case of an Overflow condition (byte / word received when both words of the receive FIFO are full) an error bit READERR in the Status Register is set. The new byte/word will not be shifted into the receive FIFO. An Under-flow condition (byte / word required to be transmitted when both words of the transmit FIFO are empty) will result in the WRITERR bit in the Status Register being set and the previous byte/word being sent.

6.3 BSIO Frequency Divider

The Frequency Divider allows the BμILD Bus clock B_CLK, to be divided down to a frequency of between B_CLK/2 to B_CLK/512, depending on the value selected by the SCLKFREQ bits in the Configuration Register. It consists of a 9-bit Synchronous Counter and SCLK Enable Logic as shown in *Figure 6.6 below*.

Two outputs are provided: SCLK_INT (the serial output clock) and SCLKX2 (twice the frequency of SCLK_INT). The frequency divider is disabled and held reset when no operation is currently in progress.

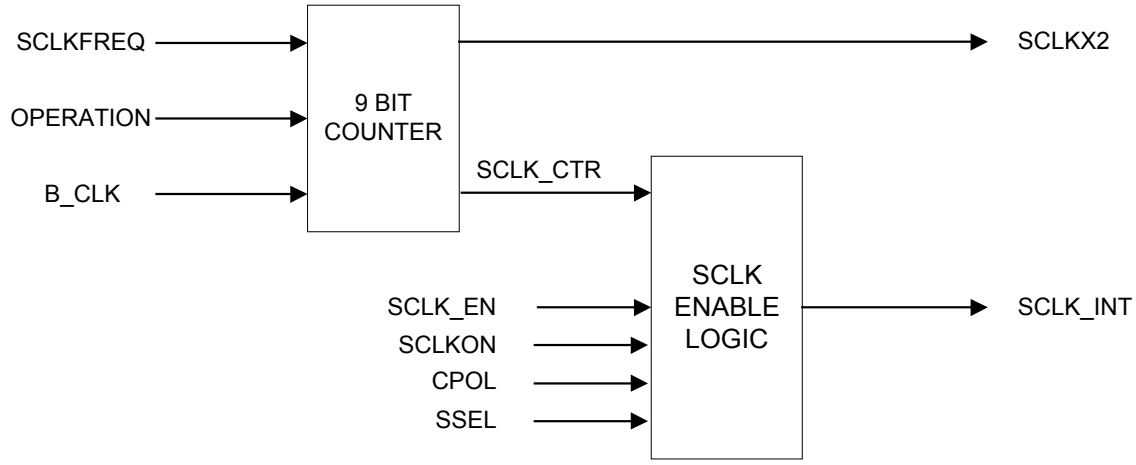


Figure 6.6 BSIO Frequency Divider

The division ratio of the counter is selected by the SCLKFREQ bits in the Configuration Register, and is in the range 2^1 to 2^9, for SCLKFREQ = 0000 to SCLKFREQ = 1000 respectively. It is clocked by the Rising Edge of B_CLK.

SCLK_EN, an output from the Sequencer, is used to enable or inhibit SCLK_INT. When SCLK_INT is in the idle state (i.e. inhibited), its polarity will be configured for each of the slaves (SS0 & SS1) by means of the CPOL bits in the Slave Select Register (High if CPOL = 1). The Timing Diagram for this is shown in *Figure 6.7 below*. Note that if there is a change in the polarity of SCLK_INT, when selecting between two devices, the start of the first operation can be delayed as required.

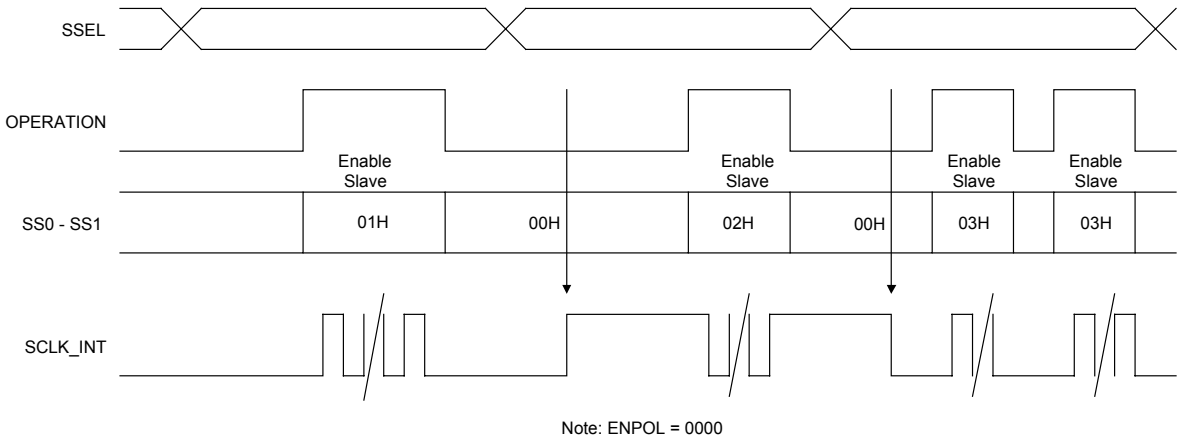


Figure 6.7 BSIO SCLK Polarity Timing

SCLKON in the Configuration Register allows SCLK_INT to be stopped during an Operation.

6.4 BSIO Slave Select Logic

The Slave Select Logic, as shown in *Figure 6.8 below*, provides the Slave enable signals. When a Slave device is to be selected, the Sequencer enables the Slave Select Logic by means of the ext_sel signal.

The SSEL bits in the Configuration Register will select either SS0_OP or SS1_OP as shown in *Table 6.1 below*:

SSEL	OP
000	SS0
001	SS1
010	Reserved
011	Reserved
100	Reserved
101	Reserved
110	Reserved
111	Reserved

Table 6.1 BSIO Slave Select Enable Configuration

In addition the ENPOL bits in the Slave Select Registers, configure the polarity of their corresponding select outputs (i.e. active High or active Low), with ENPOL = '0' selecting an active Low.

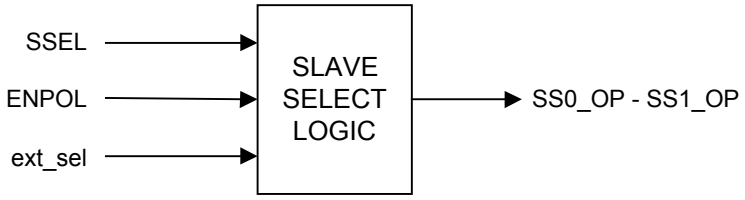


Figure 6.8 BSIO Slave Select Logic

6.5 BSIO Interrupt Control

The Active High INT output is provided to allow the BSIO to operate in an interrupt driven environment. The five interrupt sources are the WRREADY, RDREADY, WRITERR and READERR bits in the Status Register and a derivative of OPERATION, OPCOMP to denote the current operation has completed. They will be enabled by writing (Logic = High to enable) to their corresponding enable bits in the Interrupt Control Register.

6.6 BSIO Write Buffer and Control Register

The Write Buffer consists of a two 32-bit transmit FIFO and a 32-bit transmit shift register and Control Logic as shown in *Figure 6.9 below*.

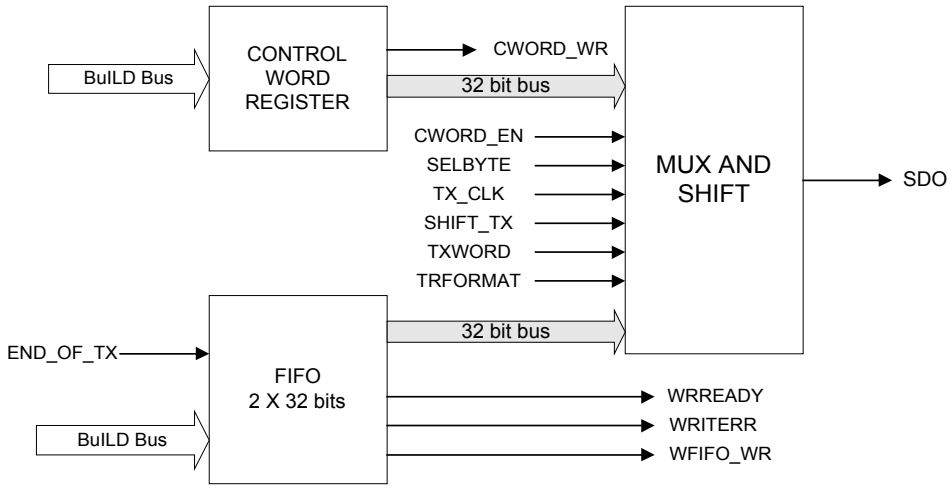


Figure 6.9 BSIO Write Buffer and Control Register

Writing to the Read/Write Buffer loads the FIFO with a 32-bit word, whose valid bits will be shifted out serially via the Transmit Shift Register. It should be borne in mind that not all the 32-bits would necessarily be sent since if byte mode were selected by SELBYTE in the Transfer Register then they would be treated as four separate bytes to be sent. However, if non-byte mode is selected then the TXWORD bits in the Transfer Register select the number of bits in a word. Note that in this case the lower order bits make up the word, with the higher order ones being redundant.

The Sequencer generates the signal TX_CLK, which forms the shift clock for serial data from the Shift Register, with SHIFT_TX being the shift enable signal. It also asserts the END_OF_TX signal, after all the valid bits within a 32-bit Word have been shifted out.

WRREADY, a bit in the Status Register is set when the FIFO is ready to receive the next word, and is cleared when the FIFO is full or when no more data is required to complete the current write operation. An Under-flow condition i.e. a byte/word to be sent when the FIFO is empty, will result in the WRITERR bit in the Status Register being set and the previous byte/word being sent. The WRITERR bit will be cleared by a read of the Status Register.

The TRFORMAT bits in the Slave Select Registers select between either a MSB or LSB first format (TRFORMAT = High selecting MSB first) for each of the six slave devices. Note that in byte mode bit 7 of the byte would be the MSB, whereas in word mode this would be its most significant bit.

The internal output WFIFO_WR will be set when the first word in an Operation is written to the FIFO, and is cleared at the end of an Operation. Whilst in Standard Mode, it will be used to set the OPERATION bit in the Status Register.

When the Sequencer asserts CWORD_EN, the Control Word is shifted out at SDO prior to any data to be written from the FIFO. CWORD_WR will be set when a Control Word is written to the Control Word Register, and will be cleared at the end of an Operation. When in Page Mode, it will set the OPERATION bit in the Status Register.

6.7 BSIO Read Buffer

The Read Buffer consists of a Receive shift register, two 32-bit receive FIFOs and Control Logic as shown in *Figure 6.10 below*.

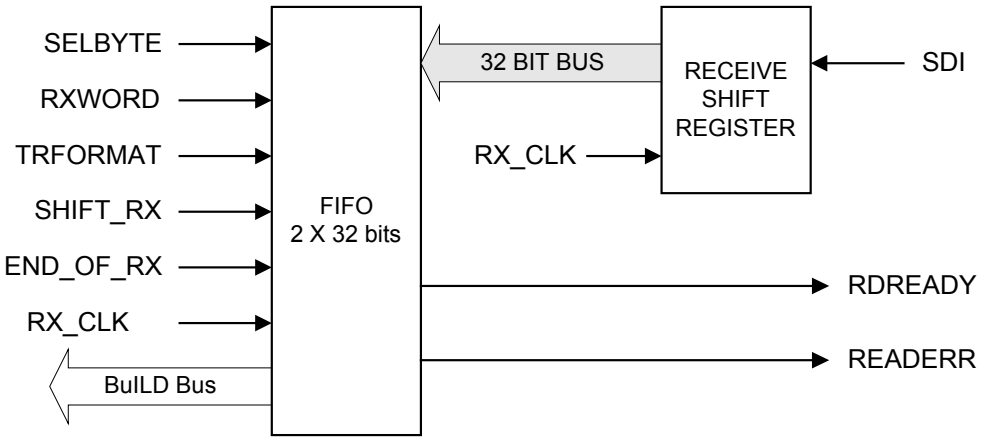


Figure 6.10 BSIO Read Buffer

Received data is shifted into the Receive shift register serially, and transferred to the FIFO in word / byte format, from where it may be read as a 32-bit word. Note that like the Write Buffer, not all 32-bits in the Read FIFO will necessarily be valid. When the SELBYTE bit in the Transfer Register selects byte mode, each 32-bit word is made up of four consecutive bytes. However in non-byte mode, the RXWORD bits in the Transfer Register set the width of the word that is to be received. Hence, the higher order bits that do not make up the word are redundant. For example if a 20-bit Word were to be selected, then bdata<19:0> would make up the word, with bdata<31:20> not being used.

Incoming serial data at SDI is shifted into the shift register by RX_CLK and the control signal SHIFT_RX both generated by the Sequencer. Once the number of valid bits that make up a 32-bit word have been transferred into the FIFO, an END_OF_RX signal generated by the Sequencer will set the RDREADY bit in the Status Register. The RDREADY bit is cleared when the Read FIFO is empty.

As the FIFO is capable of storing two 32-bit words, it is possible to store the next word before the previous one has been read.

If however a third word/byte were to be completely received whilst the FIFO is full the READERR bit in the Status Register will be set, and further data will not be stored until the first word is read. READERR is cleared by a read of the Status Register.

The TRFORMAT bits in the Slave Select Registers select between MSB and LSB formats, for each of the six slave devices. Note that in byte mode bit 7 of each byte would be the MSB, whereas in Word mode this would be its most significant bit.

6.8 BSIO Sequencer

The Sequencer consists of Read/Write Counters and Control logic as shown in *Figure 6.11 below*.

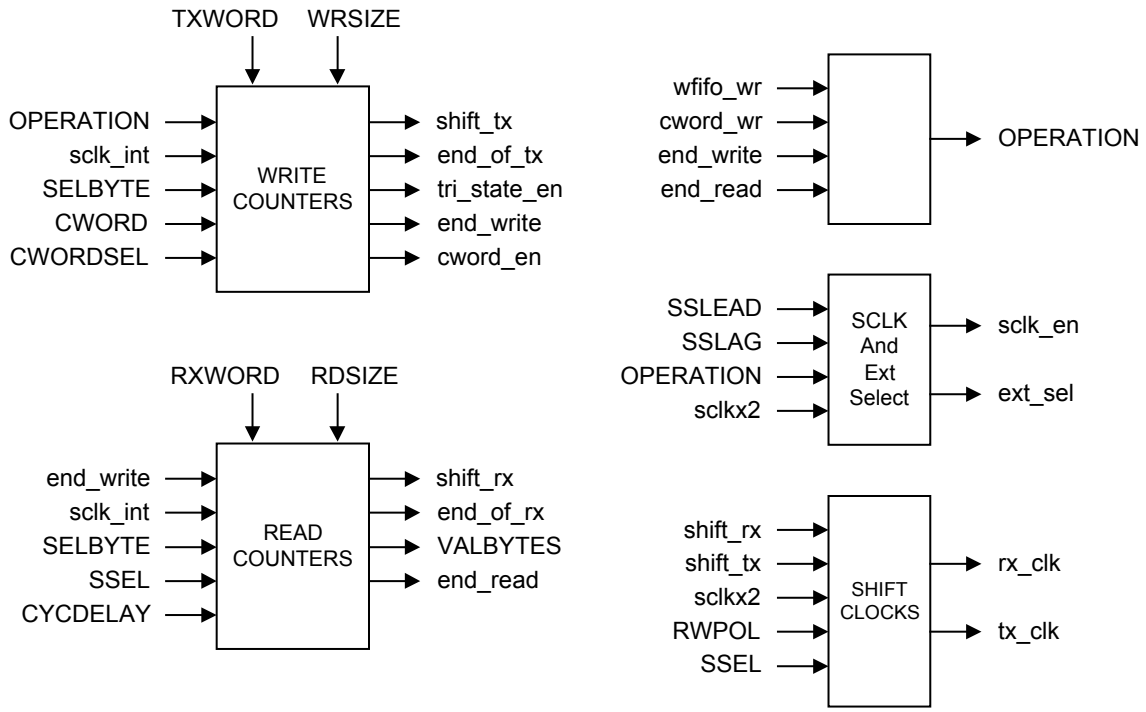


Figure 6.11 BSIO Sequencer

The Write Counters consist of a 5-bit binary counter to count the bit position within the currently transmitted control or data word. There is also a 10-bit binary down counter. This is used to count the number of data words that remain to be transmitted. At the start of an operation the counters are loaded with values determined by TXWORD / CWORD and WRSIZE in the Transfer and Mode Registers. After all the valid bits that make up a 32-bit word in the Write FIFO have been shifted, the signal End_Of_Tx is asserted. End_Write is used to start the Read Counter, after the last bit to be transmitted in the current operation has been shifted out of the Transmit Shift Register. When the SELBYTE bit in the Transfer Register is set only byte-wide operations occur. If the CWORDSEL bit in the Mode Register selects Page mode, the first word to be sent is the Control Word.

CWORD_EN acts as the shift enable signal for the Control Word Register, with CWORD in the Mode Register selecting the width of the Control Word.

The Read Counters also consist of a 5-bit binary counter to count the bit position within the currently received data word. Another 10-bit binary down counter counts the number of data words that remain to be received. At the start of an operation, the counters are loaded with values determined by RXWORD and RDSIZE in the Transfer Register. End_write, a signal from the Write Counter, is used to start the Read Counter, with CYCDELAY and SSEL being able to select a 1 cycle delay for either SS0 or SS1. Shift_Rx, the control signal to the Read Buffer is asserted when the first bit is to be read and de-asserted after the last bit has been shifted into the Receive shift register. After all the valid bits that make up a 32-bit word in the RX FIFO have been shifted in, End_Of_Rx is asserted, with end_read indicating the end of a complete read operation. The two status bits VALBYTES indicate the number of bytes received. When the SELBYTE bit in the Transfer Register is set, only byte wide operations occur.

The Operation bit in the Status Register is set when:

- the first word in an operation is written to the Write Buffer in Standard Mode,
- the Control Word is written in Page Mode.

The bit is cleared after the last bit of the current operation has been sent or received.

The Sequencer enables SCLK and the Slave Select Logic by means of SCLK_EN and ext_sel respectively. The bits SSLEAD in the Configuration Register select a delay between the external select being active and SCLK being enabled of between 1 to 4 SCLK cycles. Similarly the bits SSLAG, select the delay between SCLK being stopped and the external select being disabled of between 1 to 4 SCLK cycles.

RX_CLK and TX_CLK, the shift register clocks to the Read and Write Buffers are provided to allow data to be sent or received at either the rising or falling edge of SCLK. RDPOL, WRPOL and SSEL select the clock polarity for read and write cycles for each of the slave select outputs SS0 and SS1.

6.9 BSIO Registers

The BSIO uses nine separate registers. The GP4020 BSIO Base Address is 0xE000 7000.

Address Offset	Register	Direction	Function
0x000	*CONFIG*	Read/Write	Configuration Register
0x004	*TRSFR*	Read/Write	Transfer Control Information
0x008	*MODE*	Read/Write	Mode Register
0x00C	-	-	*Reserved*
0x010	*SLAVE0*	Read/Write	Slave Select 0
0x014	*SLAVE1*	Read/Write	Slave Select 1
0x018 to 0x02C	-	-	*Reserved*
0x030	*STATUS*	Read	Status Register
0x034	*INTC*	Read/Write	Interrupt Control
0x038	*RWBUF*	Read/Write	Read Write Buffer
0x03C	*CWBUF*	Read/Write	Control Word Buffer
0x040 to 0xFFC	-	-	*Reserved*

Table 6.2 BSIO Register Map

All registers are addressable as 32-bit locations only

6.9.1 BSIO Configuration Register - CONFIG - Memory Offset 0x0000

Bit	Mnemonic	Description	Reset Value	R/W
31:18		*Reserved*	All = 0	R
17:15	SSEL	Slave Select. Selects slave as follows: SSEL=000 SS0 SSEL=001 SS1 ALL OTHERS *Reserved*	000	R/W
14		*Reserved*	0	R/W
13		*Reserved*	0	R/W
12	CONFSDIO	Configure SDIO. A High configures SDIO as an Open Drain Output, whereas a Low configures it as a CMOS output.	0	R/W
11	CONFSCLK	Configure SCLK. A High configures SCLK as an Open Drain output, whereas a Low configures it as a CMOS output.	0	R/W

Bit	Mnemonic	Description	Reset Value	R/W
10:7	SCLKFREQ	SCLK Frequency. Select the frequency of SCLK, between B_CLK/512 to B_CLK/2 in nine increments. SCLKFREQ = 0000 selects B_CLK/2, SCLKFREQ = 0001 selects B_CLK/4, SCLKFREQ = 0010 selects B_CLK/8 Until … SCLKFREQ =1000 selects B_CLK/512. If SCLKFREQ > 1000, B_CLK/512 is selected.	0000	R/W
6:5	SSLAG	SS LAG time. Period between SCLK being stopped, and the external select being disabled. Programmable between 1 to 4 SCLK cycles, with bit settings '00' to '11' respectively.	00	R/W
4:3	SSLEAD	SS LEAD time. Period between the external select being active, and SCLK being enabled. Programmable between 1 to 4 SCLK cycles, with bit settings '00' to '11' respectively.	00	R/W
2	SCLKON	SCLK ON. When High, SCLK is ON, during an operation, when Low it is OFF.	1	R/W
1:0		*Reserved*	00	R

Table 6.3 BSIO Configuration Register

NOTE: If CONFSDIO is set High, to configure SDIO as Open Drain (i.e. Tx 0 = driven low, Tx 1 = tristate) then the SDIO pin will remain driven low if the last bit transmitted is '0'. Therefore, to do a Transmit followed by a Receive, you need to ensure that the last bit transmitted from the SDIO port on the GP4020 is a "1" (i.e. High), otherwise the SDIO will only read "0" (i.e. Low).

6.9.2 BSIO Transfer Register - TRSFR - Memory Offset 0x0004

Bit No.	Mnemonic	Description	Reset Value	R/W
31		*Reserved*	0	R
30:26	RXWORD	RX Word Width: When SELBYTE = Low, the size of the RX Word can be configured from 2- to 32-bits. RXWORD = 00000 or 00001 selects a width of 2, with RXWORD = 11111 selecting a width of 32. Note only the least significant bits selected make up the Word, with the most significant bits not used. These bits are ignored when SELBYTE = High.	11111	R/W
25:21	TXWORD	TX Word Width: When SELBYTE = Low, the size of the TX Word can be configured from 2- to 32-bits. TXWORD = 00000 or 00001 select a width of 2, with TXWORD = 11111 selecting a width of 32. Note only the least significant bits selected make up the Word, with the most significant bits not used. *These bits are ignored when SELBYTE = High.*	11111	R/W
20	SELBYTE	Select Byte. When High selects Byte Data Transfer, when Low selects Word.	1	R/W
19:10	WRSIZE	Write Size. When written configures the number of bytes/words to be sent in current operation. With WRSIZE = 0000000000 for bytes/words = 0 to WRSIZE = 1111111111 for bytes/words = 1023	00000 00000	W
	WRREM	Write Remaining. When read returns the number of bytes/words remaining to be sent in the current operation.	00000 00000	R

Bit No.	Mnemonic	Description	Reset Value	R/W
9:0	RDSIZE	Read Size. When written configures the number of bytes/words to be read in the current operation. With RDSIZE = 0000000000 for bytes/words = 0 to RDSIZE = 1111111111 for bytes/words = 1023	00000 00000	W
	RDREM	Read Remaining. When read returns number of bytes/ words remaining to be received in the current operation.	00000 00000	R

Table 6.4 BSIO Transfer Register

6.9.3 BSIO Mode Register - MODE - Memory Offset 0x0008

Bit No.	Mnemonic	Description	Reset Value	R/W
31:6		*Reserved*	All = 0	R
5	CWORDSEL	Control Word Select: When High selects Page mode, when Low selects Standard mode.	0	R/W
4:0	CWORD	Control Word Width: These bits configure the width of the Control Word between 2-bits and 32-bits. CWORD = 00000 or 00001 selects a width of 2, whilst CWORD = 11111 selects a width of 32.	11111	R/W

Table 6.5 BSIO Mode Register

6.9.4 BSIO Slave Select Registers - SLAVE0, SLAVE1 - Memory Offset (SLAVE0 = 0x0010, SLAVE1 = 0x0014)

Bit No.	Mnemonic	Description	Reset Value	R/W
31:6		*Reserved*	All = 0	R
5	TRFORMAT	Transfer Format. A High selects MSB as the first bit, with a Low selecting LSB.	0	R/W
4	CYCDELAY	Cycle Delay. Allows the insertion of a 1-cycle delay, between write and read cycles. A High selects a delay and a LOW no delay.	0	R/W
3	CPOL	Clock Polarity. When Low, the SCLK idle state is Low, when High the SCLK idle state is High. Note that if two Slaves have different idle states, then the start of an operation can be delayed as required.	0	R/W
2	WRPOL	Write Polarity. This bit selects either a rising or falling edge of SCLK for write cycles. A High selects a Rising edge, with a Low selecting a Falling edge. The edge is with respect to the BSIO block, not the slave. Hence WRPOL selecting a rising edge configures the BSIO to generate output data transitions on the rising edge of SCLK, to be latched by the current slave on the falling edge.	0	R/W
1	RDPOL	Read Polarity. This bit selects either a rising or falling edge of SCLK for read cycles. A High selects a Rising edge, with a Low selecting a Falling edge. The edge is with respect to the BSIO block, not the slave. Hence RDPOL selecting a rising edge configures the BSIO to register input data on the rising edge of SCLK, which has been generated by the current slave on the falling edge.	0	R/W
0	ENPOL	Enable Polarity. Sets the polarity of the Slave Select output as either active Low or active High. A Low configures the corresponding output as active Low, with a High configuring it as active High.	0	R/W

Table 6.6 BSIO Slave Select Register

6.9.5 BSIO Status Register - BSIO_STAT - Memory Offset - 0x0030

Bit No.	Mnemonic	Description	Reset Value	R/W
31:8		*Reserved*	All = 0	R
7	OPCOMP	Operation Complete: Set once current operation has completed. Cleared by Read of Status Reg.	0	R
6	OPERATION	Operation: Set at start of operation, by first word written to Write Buffer in Standard Mode or by writing the Control Word in Page Mode. Cleared at end of operation after last word sent or received.	0	R
5	WRITERR	Write Error: Set when an under-flow occurs in the Write Buffer. Cleared by Read of Status Register	0	R
4	READERR	Read Error. Set when an overflow occurs in the Read Buffer. Cleared by read of Status Register.	0	R
3:2	VALBYTES	Valid Bytes. Number of Valid bytes to be read, from the Read Buffer.	00	R
1	RDREADY	Read Buffer Ready. Set when Read buffer contains at least one valid word. Cleared when buffer is empty.	0	R
0	WRREADY	Write Buffer Ready. Sets whenever the Write Buffer is ready to store next 32-bit word, and more words are required to complete the current transmit operation. Cleared when buffer is full or all words have been transmitted.	0	R

Table 6.7 BSIO Status Register

6.9.6 BSIO Interrupt Control Register - INTC - Memory Offset - 0x0034

Bit No.	Mnemonic	Description	Reset Value	R/W
31:8	-	*Reserved*	All = 0	R
7	OPCOMPEN	Operation Complete Enable. Active High interrupt Enable for the OPCOMP bit in the Status Register.	0	R
6	-	*Reserved*	0	R
5	WRITERREN	Write Error Enable. Active High Interrupt Enable for the WRITERR bit in the Status Register	0	R/W
4	READERREN	Read Error Enable. Active High Interrupt Enable for the READERR bit in the Status Register	0	R/W
3:2		*Reserved*	00	R
1	RDREADYEN	Read Ready Enable. Active High Interrupt Enable for the RDREADY bit in the Status Register	0	R/W
0	WRREADYEN	Write Ready Enable. Active High Interrupt Enable for the WRREADY bit in the Status Register	0	R/W

Table 6.8 BSIO Interrupt Control Register

6.9.7 BSIO Read/Write Buffer Register - RWBUF - Memory Offset - 0x0038

Bit No.	Mnemonic	Description	Reset Value	R/W
31:0	RWBUFF	32-bit Read/Write buffer, for word to be sent and received. Data Transfer can be either byte oriented, or based on a Word Width configurable between 2- and 32-bits, for the Read and Write Buffers. When configured as a Word, only the selected number of lower order bits are employed, with the remaining higher order bits not used. This is implemented as a 2 Word by 32-bit FIFO, for both words to be sent and those received.	All = 0	R/W

Table 6.9 BSIO Read/Write Buffer Register

6.9.8 BSIO Control Word Buffer Register - CWBUF - Memory Offset - 0x003C

Bit No.	Mnemonic	Description	Reset Value	R/W
31:0	CONT	32-bit Control Word to be sent. This will be the first word sent in an operation, if the CWORDSEL bit, in the Mode Register is set. The width of this word is configurable from 2- to 32-bits via the CWORD bits in the Mode Register.	All = 0	R/W

Table 6.10 BSIO Control Word Buffer Register

7 12-CHANNEL CORRELATOR (CORR)

7.1 Introduction

The 12-Channel Correlator forms the GPS-specific module of the GP4020 GPS Baseband Processor. It comprises 12 parallel Spread-spectrum signal tracking modules, including Carrier offset mixers, C/A code generators and mixers, and 1ms Accumulate and Dump registers. *Figure 7.1 below* shows a block diagram of the correlator. It consists of the following blocks:

7.1.1 Clock Generator

The Clock Generator block divides the frequency of the 40MHz master clock, which is generated by the System Clock Generator (*refer to section 14*), by 6 or 7 to give the internal multi–phase set of clocks. The SAMPCLK pin (pin 63 (100-pin package)) is an output giving a 4:3 mark–to–space ratio clock at 40 MHz / 7 (= 5·714MHz).

7.1.2 Timebase Generator

The Timebase Generator produces three important timing signals: ACCUM_INT, TIC and MEAS_INT. ACCUM_INT is an interrupt provided to control data transfer between the correlator accumulators and the microprocessor. It may be detected by means of the ACCUM_INT output or by reading the ACCUM_STATUS_A register (where bit 15 is a flag indicating that ACCUM_INT has occurred since the previous read of this register). ACCUM_INT is cleared by reading ACCUM_STATUS_A. After power–up this interrupt occurs every 505.05μs. The Interrupt period can subsequently be changed by:

- toggling the INTERRUPT_PERIOD bit of the SYSTEM_SETUP register, or
- writing directly to the PROG_ACCUM_INT register.

TIC is an internal signal with a default period of 99999.90μs. It is used to latch measurement data (Epoch count, Code phase, Code DCO phase, Carrier DCO phase and Carrier cycle count) of all 12 channels at the same instant. Its period can subsequently be changed, by writing to the PROG_TIC_HIGH and PROG_TIC_LOW registers, or toggling the FRONT_END_MODE bit of the SYSTEM_SETUP register.

MEAS_INT is a signal derived from the TIC counter. It may be used by the microprocessor as a software module switching interrupt either by using the MEAS_INT output or by reading the ACCUM_STATUS_B or MEAS_STATUS_A registers. MEAS_INT is activated at each TIC and 50 ms before each TIC provided the TIC period is greater than 50 ms. If the TIC period is less than 50 ms, MEAS_INT is activated only at each TIC. It is cleared by reading either the ACCUM_STATUS_B or MEAS_STATUS_A register, depending upon the MEAS_INT_SOURCE bit of the SYSTEM_SETUP register.

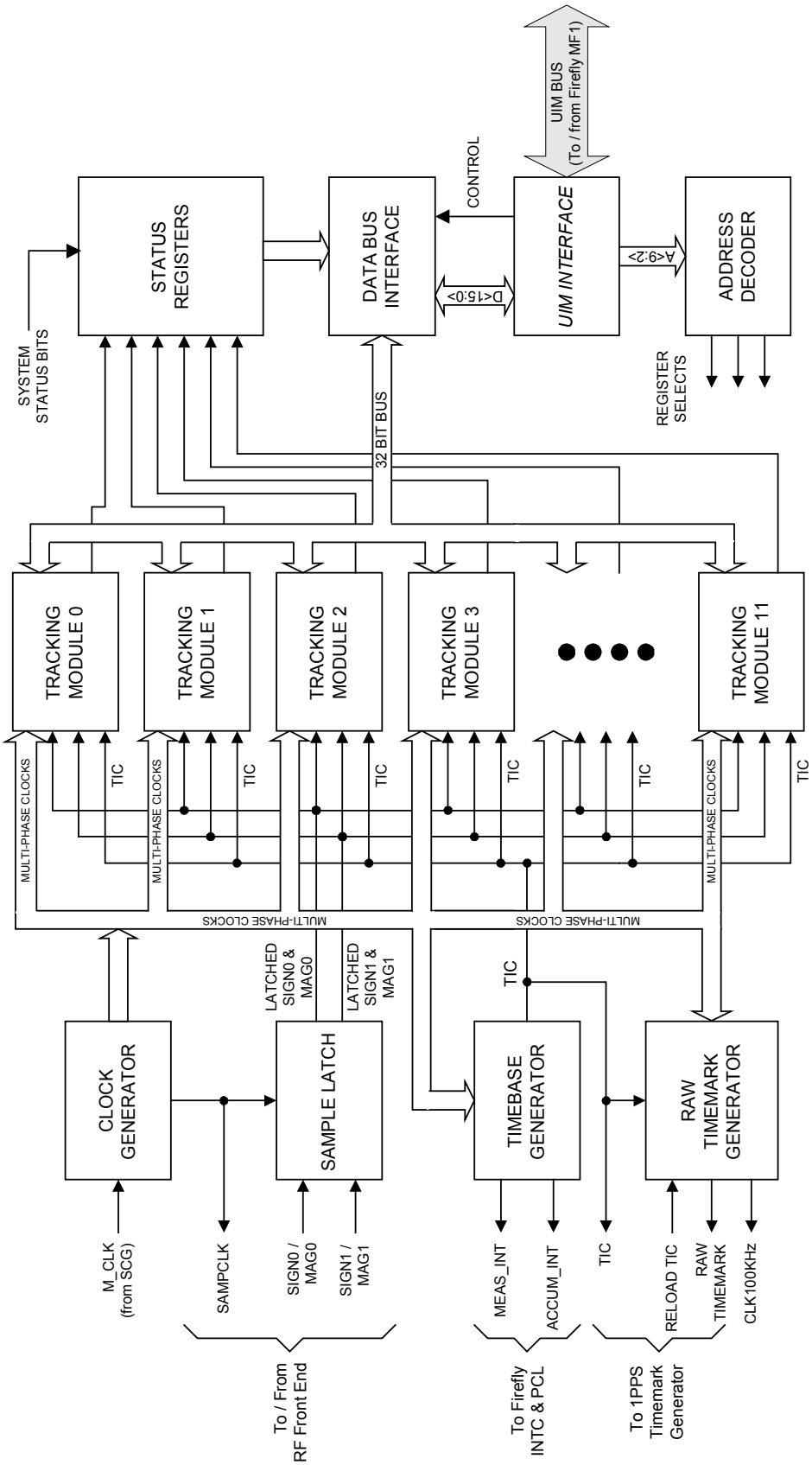


Figure 7.1 12-Channel Correlator Block Diagram

7.1.3 Raw-Timemark Generator

The Raw Timemark generator generates two essential signals:

1) **CK100kHz.** This 100kHz clock is derived from M_CLK. This clock is only as accurate as the receiver TCXO attached to the RF front-end, and hence cannot be re-synchronised to any GPS system timing. This signal can be accessed from the Peripheral Control Logic (PCL) block.

2) **Raw Timemark**. This is a one Pulse-Per-Second (1PPS) signal which is derived the Multiphase Clock and the TIC signal from the Timebase generator.

The Raw Timemark signal can be aligned to UTC (Universal Time Co-ordinated) by means of dynamic software skewing of the TIC period within the Timebase Generator. The resolution of the TIC skewing is limited to 175ns minimum. Improved alignment resolution can be achieved in conjunction with an external block of circuitry, the 1PPS Timemark Generator.

There are two methods of generating the Raw Timemark signal. In both cases TIMEMARK rising edges are generated coincident with the rising edges of TIC. Therefore, for TIMEMARK to be aligned with UTC, TIC must be aligned with UTC. This can be done by modifying the TIC period for a single TIC cycle, then setting it back to its original value, thus slewing the phase of TIC. The alignment to UTC can also be performed independently of TIC slewing by an independent delay counter within the external 1PPS Timemark generator.

Raw Timemark can be generated using two methods:

1) Set "FREE_RUN_TIMEMARK" (Bit 1 in TIMEMARK_CONTROL register) to ‘0’ and by setting the ARM_TIMEMARK bit (Bit 0); the next TIC will generate a rising edge at RAW_TIMEMARK and clear the ARM_TIMEMARK bit.

2) Set "FREE_RUN_TIMEMARK" to '1', and set the number of TICs per RAW_TIMEMARK period in the "FREE_RUN_RATIO" bits of the register (Bits 2 to 6). In this instance, a RAW_TIMEMARK output pulse is generated automatically every "FREE_RUN_RATIO" TICs automatically.

Further details of producing 1PPS Timemark are covered within *Section 15 "1PPS TIMEMARK GENERATOR" on page 149* for more information.

7.1.4 Status Registers

There are four status registers (ACCUM_STATUS_A, _B, _C and MEAS_STATUS_A). These contain flags associated with the accumulated and measurement data held on each of the 12 channels. Some system level status bits also appear in these registers.

7.1.5 Sample Latches

The Sample Latches synchronise data from the front end to the internal SAMPCLK. The down converted satellite signal can be sampled at the output of the front end by SAMPCLK. This data is then input to the 12-channel correlator as 2-bit data on the SIGN0, MAG0, where it is resampled at the next rising edge of SAMPCLK. These signals are then distributed to the 12 tracking modules. When a GP2015 or GP2010 front end is used, the data represents a band–limited signal at an IF centred on 4.309MHz. The IFOUT will alias to a 1.405MHz digitised IF, by sampling the 4.309MHz signal at a rate of 5.714MHz.

7.1.6 Address Decoder

The Address Decoder performs address decoding for the correlator.

7.1.7 Bus Interface

The Bus Interface controls the transfer of data between the external 16-bit wide data bus and the internal 32-bit data bus. Apart from the code and carrier DCO increment values, all data transfers are 16-bits wide. Write

operations to the code and carrier DCO's are 32-bit data transfers, in which the High 16-bit word must be written immediately before the low 16-bit word. Note that the write cycle to write cycle delay of 300ns referred to in the Microprocessor Interface does not apply between the first and second write cycles for 32-bit DCO data transfers. For further information, refer to *Section 7.5 "12 Channel Correlator Interface Timing" on page 63.*

7.1.8 UIM Interface

The UIM Interface performs data conversion and re-timing between the 12-channel correlator (clocked with the M_CLK signal from the System Clock Generator), and the Up Integration Module Bus (UIM bus), clocked by BμILD_CLK. For further information, refer to *Section 11 "MEMORY PERIPHERAL CONTROLLER (MPC)" on page 109*.

7.2 Tracking Modules

The Tracking Modules are 12 identical signal-tracking channels numbered CH0 to CH11, each with the block diagram shown in *Figure 7.2 below*. These blocks generate the data used to track the satellite signals. There is no overwrite-protection mechanism on this data. For further information, refer to *Section 7.4 "Controlling the 12 Channel Correlator" on page 59.*

Each Tracking Channel can be individually programmed to operate in either UPDATE or PRESET mode. Update mode is the normal mode of operation. PRESET mode is a special mode of operation where writes to certain registers are delayed until the next TIC to allow synchronisation of registers and pre-setting of the code DCO phase. For further information, refer to *Section 7.4.9 "PRESET Mode" on page 61.*

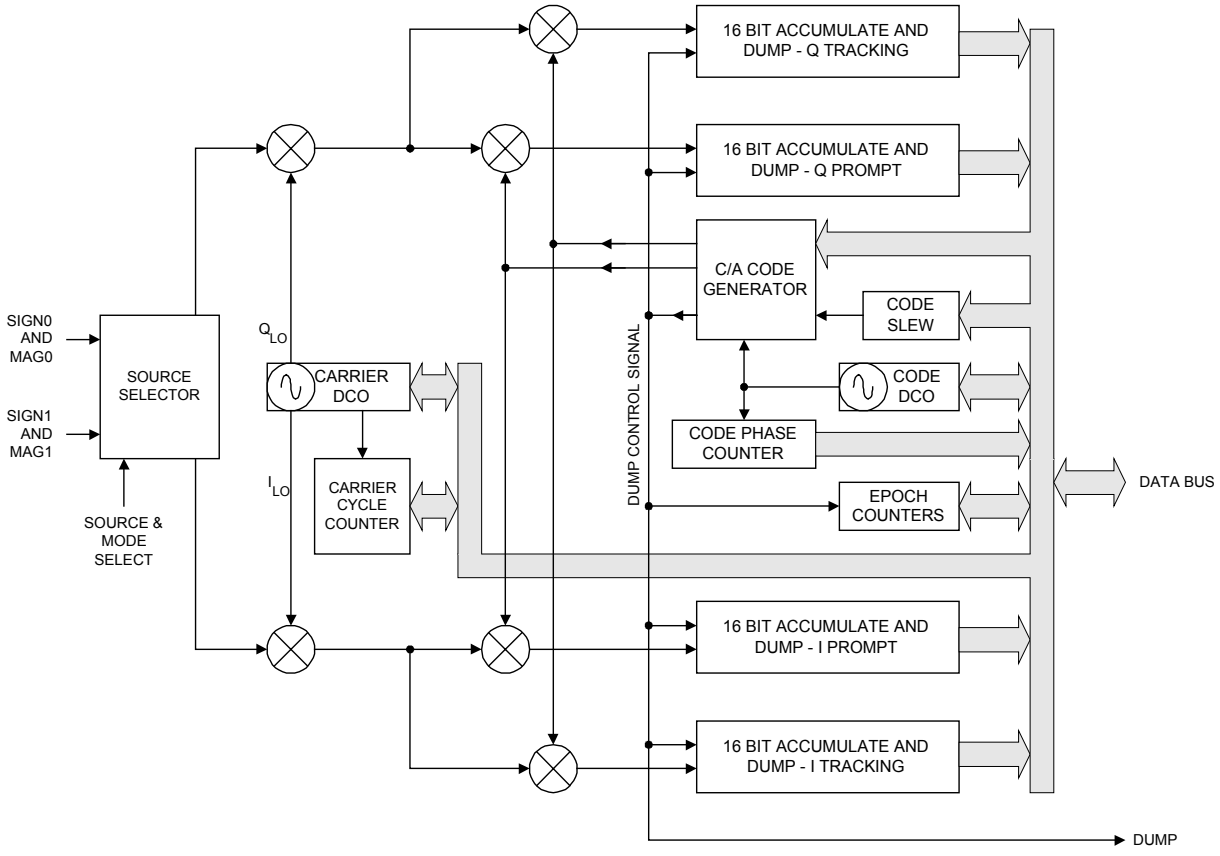


Figure 7.2 Tracking Module Block Diagram

The individual sub–blocks in the tracking modules are:

7.2.1 Carrier DCO

The Carrier DCO, which is clocked at the SAMPCLK frequency, is used to synthesise the digital Local Oscillator signal required to bring the input signal to baseband in the mixer block. It must be adjusted away from its nominal value to allow for Doppler shift and reference frequency error.

When used with the GP2015/GP2010 the nominal frequency of this signal is 1.405396825 MHz (with a resolution of 42.57475 MHz) and is set by loading the 26-bit register CHx_CARRIER_DCO_INCR. This very fine resolution is needed so that the DCO will stay in phase with the satellite signal for an adequate time.

The Carrier DCO Phase cannot be directly set, but must be adjusted by altering the frequency. The Carrier DCO outputs are 4 level, 8 phase sinusoids with the following sequences over one cycle:

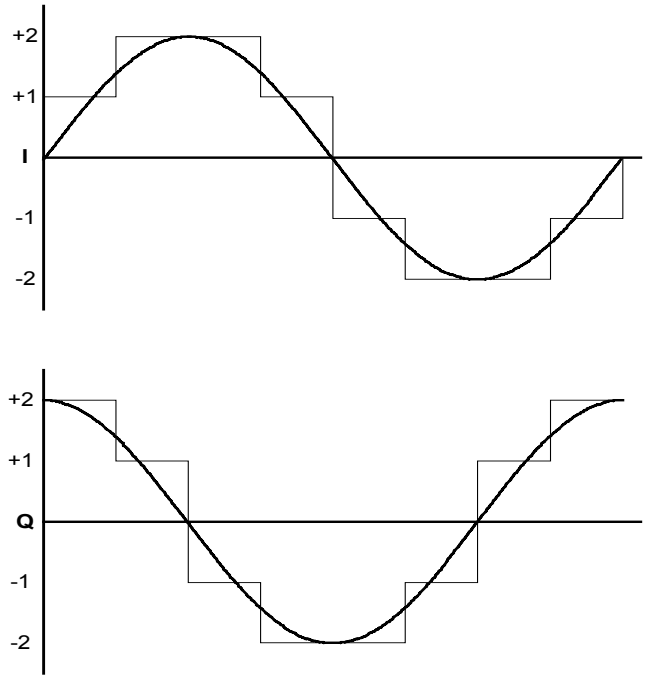


Figure 7.3 Waveform outputs from Carrier DCO I & Q LO (*sinewaves are a guide only*)

Destination Arm	Sequence
I_{LO}	-1 +1 +2 +2 +1 -1 -2 -2
Q_{LO}	+2 +2 +1 -1 -2 -2 -1 +1

Table 7.1 12-channel Correlator Carrier DCO outputs

As the clock to the DCO is normally less than 8 times the output frequency, not all phases are generated in every cycle. With a typical clock frequency of 5.714 MHz and an output frequency of 1.405 MHz, there are only approx. 4 phases per cycle. These will slide through the cycle as time progresses to cover all values.

7.2.2 Code DCO

The Code DCO is similar to the Carrier DCO block. It is also clocked at the SAMPCLK frequency, and synthesises the oscillator required to drive the code generator at twice the required chipping rate. The nominal frequency of the

output is 2.046 MHz, to give a chip rate of 1.023 MHz and is set by loading the 25-bit register CHx_CODE_DCO_INCR.

It is programmed with a resolution of 85·14949 mHz when used with a GP2015/GP2010 front end. The very fine resolution is again needed to keep the DCO in phase with the satellite signal. The Code DCO Phase can only be set to the exact satellite phase in PRESET mode. In Update mode, it must be aligned with the satellite phase by adjusting its frequency.

7.2.3 Carrier Cycle Counter

The Carrier Cycle Counter is 20-bits long, and keeps a count of the number of cycles of the Carrier DCO between TICs. This is not needed for a basic navigation system but may be used to measure the range change (delta–range) to each satellite between TICs. The delta ranges can be used to smooth the code pseudo–ranges. For finer detail the Carrier DCO phase may also be read at each TIC to give the fractional part of the cycle count or delta–range.

7.2.4 C/A Code Generator

The C/A Code Generator generates the selected Gold code for:

i) a GPS satellite (1 to 32),
ii) a ground transmitter (pseudolite, 33 to 37),
iii) an INMARSAT–GIC satellite (201 to 211),
iv) an INMARSAT-WAAS satellite (120 to 138)
v) a GLONASS satellite.

A Gold code is selected by writing a specific pattern of 10-bits to the CHx_SATCNTL register, or by setting the GPS_NGLON bit to Low for the GLONASS code. Two outputs are generated to give both a PROMPT and a TRACKING signal. The TRACKING signal can be set to one of four modes: EARLY (one half chip before the PROMPT signal), LATE (one half chip behind), DITHERED (toggled between EARLY and LATE every 20ms) or EARLY–MINUS–LATE (the signed difference). The output code is a sequence of +1's and –1's for all code types except EARLY–MINUS–LATE where the result can also be a '0'. To avoid having an unused LSB in the accumulators, the values in EARLY–MINUS–LATE mode are halved from the +2, 0, –2 that results from the calculation (+1 or –1) – (+1 or –1) to +1, 0, –1. This must be considered when choosing thresholds in the software, as the correlation results will be exactly half of the values otherwise expected.

At the end of every code sequence (1023 chips in GPS mode or 511 chips in GLONASS mode), a DUMP signal is generated to latch the Accumulated Data for use by the signal tracking software. Each channel is latched separately, as the satellite signals are not received in phase with each other.

The nature of GLONASS signals is that they are modulated with the same PRN Gold Code, but are separated in the frequency domain (1597MHz to 1617MHz). Navstar GPS signals are modulated with different PRN Gold Codes, but are transmitted on the same frequency (L1 = 1575.42MHz).

For the GP4020 to effectively de-modulate GLONASS signals, it would ideally need to have a separate set of RF signal inputs for each correlator channel, in order for it to differentiate between the different frequencies used by each GLONASS satellite. Since this facility is NOT available, the GP4020 cannot be used effectively to decode a constellation of GLONASS signals.

Although the GP4020 contains a C/A code generator that can be used to demodulate GLONASS signals, the GP4020 does NOT have a sufficiently wide Doppler-offset compensation range to allow it to be used effectively for GLONASS. (The GLONASS code can be selected by setting the GPS_NGLON bit in CHx_SATCNTL to '0'). The total frequency offset, which the carrier DCO can cope with, is ±2.857MHz. This means that the GP4020 12-channel correlator will NOT be able to deal with the complete range of GLONASS signals, unless they are mixed separately down to a digital IF of approx. 1.4MHz.

The GP1020, also available from Zarlink Semiconductor, has 10 separate sets of SIGN / MAG inputs from RF front-end devices, which can be configured to connect independent to any of 6 correlator channels, making this device more suitable for GLONASS applications. Contact your regional Zarlink sales office for more information.

7.2.5 Carrier Mixers

The Carrier Mixers multiply the digital input signal by the Carrier DCO digital local oscillator to generate a signal at baseband. Both the I and Q Carrier DCO phases are directed to the appropriate mixers. The mixing of the Carrier DCO outputs with the input signal that produces a baseband signal, which can have the values ±1, ±2, ±3 and ±6.

7.2.6 Code Mixers

The Code Mixers multiply the I and Q baseband signals from the Carrier Mixers with both the PROMPT and TRACKING local replica codes to produce four separate correlation results. The correlation results are passed to the Accumulate and Dump blocks for integration.

7.2.7 Accumulate and Dump

The Accumulate and Dump blocks integrate the Mixer outputs over a complete code period of nominally 1ms. There are four separate 16-bit accumulators for each channel. These represent the correlation of the I and Q signals with the PROMPT and TRACKING codes, over the integration period. There is no overwrite protection mechanism on these registers so the data must be read before the next DUMP.

7.2.8 Code Phase Counter

The Code Phase Counter counts the number of half–chips of generated code and stores this value in the CHx_CODE_PHASE register on each TIC.

7.2.9 Code Slew Counter

The Code Slew Counter is used to slew the generated code by a number of half chips in the range 0 to 2047. In Update mode, the slew occurs following the next DUMP. In PRESET mode, it occurs at the next TIC. All slew operations are relative to the current code phase. The Code Slew counter must be written to, each time a slew is required. During the slewing process, the accumulators for the channel being slewed are inhibited so that the first result is valid. If a slew is written while a channel is disabled, the slew will occur as soon as the channel is enabled.

7.2.10 Epoch Counter

The Epoch Counters keep track of the number of code periods over a 1-second interval. This is represented by a 5-bit word for the number of 1 ms integration periods (0 to 19), plus a 6-bit word containing the number of 20 ms counts (0 to 49). The Epoch Counters can be pre-loaded to synchronise them to the data stream coming from the satellite. This value will be transferred immediately to the counter when in Update Mode, or after the next TIC if in PRESET Mode.

The Epoch Counter values are latched on each TIC into the CHx_EPOCH register. In addition, the instantaneous values are available from the CHx_EPOCH_CHECK register.

7.3 Software Requirements

The GP4020 12-channel correlator can be operated in different ways dependent upon the GPS Receiver system required by the user. So to accommodate this and to allow dynamic adjustment of loop parameters, the GP4020 12-channel correlator has been designed to use software for as many functions as possible. This flexibility means that the device cannot be used without a microprocessor closely linked to it. Since a processor is always needed to convert the output of the 12-channel correlator into useful information, this is not a significant limitation.

The software associated with the 12-channel correlator can be divided into two separate modules:

1) Acquire and track satellite signals to give pseudo-ranges;

2) Process pseudo-ranges to give the navigation solution and format it in a form suitable for the user.

In order for a Navigation Solution to be achieved, all of the pseudo-ranges must have exactly the same clock error. This clock error can then be removed iteratively to give real ranges if sufficient satellites are tracked (three if the height is known, otherwise four). This need for exact matching of timing errors explains the need for all of the complicated synchronisation between all 12-channels of the correlator.

The following relates only to the signal processing aspects of the software, to acquire and track signals from up to twelve satellites and to obtain the pseudo-ranges and the navigation message. The operation of the navigation software is not dependent on the details of the correlator, and so does not need to be included in this data sheet.

A pair of on-chip interrupt time-base signals is provided to help implement a data transfer protocol between the microprocessor and the 12-channel correlator at fixed time intervals:

1) **ACCUM_INT**. Used to interrupt the microprocessor to retrieve Accumulated data (1.023ms worth) - period of interrupt normally less than 1ms.

2) **MEAS_INT**. Used to interrupt the microprocessor to retrieve Measurement data that occurs once every TIC (approx. 100ms period).

These interrupts can be used to achieve instant response from the microprocessor via an Interrupt Service Routine. Otherwise software-based polling will be needed; the choice is set by the application. If the ACCUM_INT interrupt is used, and perhaps also if polling is used, the data transfer rate is about twice the correlation result rate for each channel, so many transfers will not give new data. Bus use can be reduced by examining the status registers before each transfer to see if new data is available and then only reading the data if it is useful.

It is important to note that the timing of each of the correlator channels will be locked to its own incoming signal and not to each other or to the microprocessor interrupts, so new data is generated asynchronously. The sampling instant of measurement data of all channels however is common to give a consistent navigation solution.

In order to acquire lock to the satellites as quickly as possible, the data from the last fix should be stored as a starting point for the next fix. It is also useful to make use of the embedded real-time clock on the chip to give a good estimate of GPS time for the next fix. The navigation solution can be used to measure clock drift and calculate a correction for the clock to overcome ageing. The user's location (or a good estimate of it) along with the Almanac and the correct time will indicate which satellites should be searched for. These may be used to find an estimate of Doppler effects, while the previous clock error is the best available estimate of the present clock error. If this information is not available then the receiver must scan a much wider range of values, which will greatly increase the time to lock. The satellite Clock Correction and Ephemeris are needed for the navigation solution. If a recent set of this information is held in memory, the calculations may begin as soon as lock is achieved and not need to wait for the Satellite Navigation message re-transmission (18 to 36 seconds).

The 12-channel correlator on the GP4020 contains four different types of registers:

- **Control Registers** that are used to program functions of the device.
- **Status Registers** that provide a status indication of the process taking place in the device.
- **Accumulated Data Registers** that provide the results of correlation with the C/A code every millisecond. This is the raw data used to acquire and track satellite signals.
- **Measurement Data Registers** which latch the carrier DCO phase, carrier cycle count, code DCO phase, 1 millisecond epoch, and the 20 millisecond epoch count at every 9.09 or 100 milliseconds interval. This is the raw data used to compute pseudorange.

7.3.1 Software Sequence For Acquisition

The spectrum of each Navstar GPS satellite signal is spread-spectrum modulated using 1023 chip Gold codes. This causes the satellite signals seen by a GPS receiver to be so weak that they are buried in the noise and can only be detected by correlation. The GPS signal power at a GPS receiver antenna is typically approx. -130dBm, whereas the noise in the GPS signal band (2.046MHz wide) is approx. -111dBm at room temperature. To correlate

the received signals therefore, a locally generated code must be chosen to precisely match the spreading code type, rate, and phase.

This pattern is then multiplied bit-by-bit with the incoming data stream and the results integrated over the code length to recover the signal.

The process of signal acquisition is simply the matching of receiver settings to the actual signal values. To make matters more complicated the satellite carrier frequency is shifted a little by the Doppler effect due to the motion of the satellite. The user clock will drift randomly, and (in most situations) the signal-to-noise-ratio (SNR) is poor for some satellites. Therefore, the software must be 'wide-band' to find the signal and 'narrow-band' to reduce noise, leading to very different programs in different applications. For all tracking channels, the signal processing software needs the following sequence of activities:

1. **Program CHx_SATCNTL register** to select the desired GPS Gold code (PRN number) for the selected satellite and also select the code type for the mode of the correlator tracking arm. It is often best when in acquisition mode to fix the tracking arm to Dithering Mode (alternate 'Early' and 'Late') and do a search in two phases at once. Then switch the tracking arm to a tracking mode once a satellite is found.

2. **Program CHx_CARRIER_DCO_INCR_HIGH and CHx_CARRIER_DCO_INCR_LOW registers.** The values programmed into these two registers are concatenated and are used to set the local oscillator frequency for the digital mixing performed in the 12-channel correlator. This brings the incoming 2-bit digitised signal from the RF Front-end down to baseband. The value to be programmed is equal to the nominal local oscillator frequency plus the estimated Doppler shift compensation plus the estimated user clock frequency drift compensation.

3. **Program CHx_CODE_INCR_LO and CHx_CODE_INCR_HI.** The value to be programmed in these registers represents twice the nominal chipping rate of the C/A code (2.046 MHz); in addition, if desired, a small compensation for the Doppler shift and user clock frequency drift.

4. **Release the tracking channel reset** by programming the RESET_CONTROL register with the proper value. This will cause the correlation process to start.

5. **Obtain accumulated data** from Accumulated Data Register readings. Several consecutive readings on the same tracking channel can be added to increase, at will, the integration period of the correlation.

6. **Decide if the GPS signal has been found** by comparing the correlation result with a threshold. If found then jump to a signal pull-in algorithm. Note that both in-phase and phase quadrature accumulated data have to be considered since at this time, the carrier DCO local oscillator phase is not necessarily in phase with the incoming GPS signal.

7. **If the GPS signal has not been found,** a new trial has to be made with different carrier DCO, code DCO, or gold code phase programming. Typically, both DCOs would be held constant while the Gold code phase is varied to try all of the 2046 half chip positions possible. The carrier DCO would then be programmed with slightly different values and the Gold code phase positions would again be scanned. The Gold code phase is varied by programming the CHx_CODE_SLEW register and can be varied by increments of half a code chip.

8. **Once the GPS signal has been found,** the code phase alignment, the carrier phase alignment and the Doppler and user clock bias compensations are still coarse. The code phase alignment is only within a half code chip, the carrier DCO is not in phase with the incoming signal and infrequency is still in error by up to the increment used for successive trials.

The signal processing software must next use a pull-in algorithm to refine these alignments. There are many suitable types of algorithm to choose from, such as:

i) successive small steps until the error is too small to matter, like an analogue PLL;

ii) using more complicated signal processing to estimate the errors and jump to a much better set of values.

The signal pull-in algorithm will then program CHx_CARR_INCR_LO / HI registers with more values that are accurate for the Carrier DCO. Corrections to the Gold code phase smaller than a half chip cannot be done by programming CHx_CODE_SLEW registers in the Code Generator. This can be done by setting the

CHX_CODE_INCR_LO / _HI registers to steer the Code DCO and gradually bring the gold code phase to the right value.

7.3.2 Signal Tracking

The incoming GPS signal will exhibit a Doppler shift that varies with time due to the non-uniform motion of the satellite relative to the receiver, and the user clock bias is likely to also vary with time. The net result is that unless dynamic corrections are applied to the code and carrier DCOs, the GPS signal will be lost. This leads to two servo loops being required: one to maintain lock on the Gold code phase and a second to maintain lock on the carrier. This can be implemented with the following.

The raw data used to steer the two servo loops is the Accumulated Data, which is output by the tracking channel, at the rate of once per millisecond. The tracking arm Accumulated Data is used for the Gold code loop. Some approaches use an 'early minus late' Gold code to implement a null steering loop. Others use a dithering code that alternates between a code one-half chip late and a code one-half chip early. In the GP4020 12-channel correlator, the dithering rate is 20 ms (20 code epochs) each way, starting with Early after a reset, when this type of code is selected through the CHx_CNTL register. The Gold code loop is closed by regularly updating the code DCO frequency using the CHx_CODE_INCR_LO / _HI registers.

The prompt arm Accumulated Data is used for the carrier phase loop (although the dithering mode in the tracking arm may also be used). One approach consists of varying the carrier DCO phase in order to maintain all the correlation energy in the in-phase correlator arm and none in the phase-quadrature correlator arm. The carrier phase loop is closed by regularly updating the carrier DCO frequency using the CHx_CARR_INCR_LO / _HI registers.

7.3.3 Data Demodulation

The C/A code is modulated with Space Vehicle (SV) data at 50 Baud to give the navigation message. This modulation is an exclusive-OR function of the C/A code with the SV data. This means that every 20 milliseconds, (which is every 20 C/A code epochs); the C/A code phase will be reversed (shifted by 180 degrees) if the new data bit is different from the previous one. On the prompt arm, once the signal is being correctly tracked, such a data bit transition will change the sign of the accumulated data. Data demodulation can then be achieved in two stages:

1. Locate the instants of data bit transitions to identify which C/A code epoch corresponds to the beginning of a new data bit. This will allow initialisation of the 12-channel correlator epoch counters by the signal processing software (through the CHx_1MS_EPOCH and CHx_20MS_EPOCH registers) to count code epochs from 0 to 19 in phase with data bits. At each new cycle of the 1 ms epoch counter, the 20 ms epoch counter will increment.

2. Record the sign of accumulated data on the prompt arm for each data bit period of 20 ms, with filtering to reduce the effect of noise on the signal. Note that there is a sign ambiguity in the demodulation process in that it is not possible to tell which data bits are '0's and which are '1's from the signal itself. This ambiguity will be resolved at a later stage when the full Navigation Message is interpreted.

7.3.4 Pseudorange Measurement

The measurement data registers provide the raw data necessary to compute the pseudorange. This raw data is a sample, at a given instant set by the TIC signal, of the 20 ms and 1 ms epoch counters, the C/A code phase counter and the code DCO phase. By definition, the pseudo-range is expressed in time units and is equal to the satellite-to-receiver propagation delay plus the user clock bias. The user clock bias is first estimated (blind guessed is more likely with a cold start, but iteration then takes longer) and then obtained as a by-product of the navigation solution. The pseudorange is equal to the user's apparent local time of reception of the signal (t_1) minus the GPS real time of transmission (t_2).

With the demodulated data, the software has access to the Space Vehicle Navigation Message, which contains information on the GPS system time for the transmission of the current sub-frame; this is equal to term t_2.

The time information in the navigation message allows the receiver time to be initialised with a resolution of 20 milliseconds (one data bit period) but with knowledge of the precision to much better than one C/A code chip; a little less than 1 microsecond. As the time-of-flight from the satellite to the receiver is in the region of 60 to 80

milliseconds, an improved first guess for local time could include an allowance for this delay to reduce the iteration time later.

By using the data to time-tag the TIC, along with the values of the Epoch counter, the Code generator phase, and the Code clock phase it is possible to measure the time of the SV signal in local apparent time. This gives the value of t_1 needed for the pseudo-range measurement. The pseudorange can now be computed as $t_1 - t_2$.

The error present in the time setting is the initial value of the user clock bias, with an allowance for the various counter phases. Once a Navigation Solution has been found the clock error is precisely known and may be used for future pseudorange calculations. Due to the receiver clock drifting with time, the clock-bias changes with time, and this must be tracked by the Navigation software.

7.4 Controlling the 12 Channel Correlator

The following section describes typical methods for controlling the 12-channel correlator block in the GP4020. These include signal acquisition tracking and carrier phase measurement.

7.4.1 Search Operation

To perform signal acquisition, the carrier frequency and code phase space needs to be searched until the signal is detected. The maximum carrier frequency excursion from its nominal value is defined by the maximum carrier Doppler shift plus the maximum receiver clock error. The maximum code phase is defined by the (fixed) code length. Typically, all code phases will be searched at a given carrier frequency before advancing to the next carrier frequency bin and repeating the code phase search.

7.4.2 Carrier DCO Programming

The CHx_CARRIER_DCO_INCR_HIGH (or X_DCO _INCR_HIGH), and CHx_CARRIER_DCO_INCR_LOW registers are programmed in sequence with the relevant data according to the frequency bin being searched. It is always necessary to write to both the _HIGH and _LOW registers. Carrier DCO programming will become effective as soon as the channel is released (made active). If the channel is already active, writes to CHx_CARRIER_DCO_INCR_LOW are effective immediately. (A small delay of up to 175ns will occur, to allow synchronisation of the processor write operation to the chip operation.)

7.4.3 Code DCO Programming

The CHx_CODE_DCO_INCR_HIGH (or X_DCO_INCR_HIGH), and the CHx_CODE_DCO_INCR_LOW registers are programmed in sequence with the relevant data according to the estimated code frequency offset. It is always necessary to write to both _HIGH and _LOW registers. Code DCO programming will become effective as soon as the channel is released (made active). If the channel is already active, writes to CHx_CODE_DCO_INCR_LOW are effective immediately. (A small delay of up to 175ns will occur to allow synchronisation of the processor write operation to the chip operation).

7.4.4 Code Generator Programming

For each channel, the CHx_SATCNTL register is programmed as follows:

i. Set the TRACK_SEL bits to set the Tracking arm code to either early or late (with respect to the Prompt arm).

ii. Set the G2_LOAD bits to select the required PRN code.

iii. Program the CHx_CODE_SLEW register with the desired code phase offset. The slew operation will become effective upon CHx_RSTB release. The first DUMP will generate accumulated data for the channel and set the associated CHx_NEW_ACCUM_DATA status bit.

iv. Release the relevant CHx_RSTB bits of the RESET_CONTROL register to make the channel active.

When the code clock is inhibited (to slew the code phase), the Integrate and Dump module is held at reset. It will start to accumulate correlation results only after the slew operation is completed.

A search for a satellite on more than one channel may be performed using the MULTI channel addresses and different code slew values as appropriate.

7.4.5 Reading the Accumulated Data

At each DUMP, the corresponding CHx_NEW_ACCUM_DATA status bit is set in the ACCUM_STATUS_A register. The status register, together with all accumulation registers (CHx_I_TRACK, CHx_Q_TRACK, CHx_I_PROMPT, CHx_Q_PROMPT) are mapped into consecutive addresses. These can be read as a consecutive block, if required, after every ACCUM_INT interrupt. Alternatively, the Status Registers may be polled. The Accumulation registers are not overwrite-protected; therefore the system must respond quickly when new data becomes available. Whether or not it is necessary to process the accumulation at every DUMP is dependent upon the application. The order of reading them is optional, but ideally, the CHx_Q_PROMPT register should be read last, because this resets the CHx_NEW_ACCUM_DATA bit. The CHx_MISSED_ACCUM bits in the ACCUM_STATUS_B register indicate that new accumulated data has been missed. These can only be cleared by a write to CHx_ACCUM_RESET or by deactivating the channel.

7.4.6 Search on Other Code Phases

When it is desired to correlate on the next code phase, such as one whole chip later, the CODE_SLEW has to be programmed with a value of 2 (the units are half code chips). The slew will occur on the next DUMP. The effect of CODE_SLEW is relative to the current code phase. To repeat a CODE_SLEW, the register needs to be written to again even if the same size slew is required.

Once the signal has been detected (correlation threshold exceeded), the code and carrier tracking loops can be closed.

The tracking loop parameters must be tailored in the software to suit the application.

7.4.7 Data Bit Synchronisation

The data bit synchronisation algorithm should find the data bit transition instant. The processor calculates the present one-millisecond epoch and programs this value into the 1MS_EPOCH counter. Ideally, epoch counter accesses should occur following the reading of the accumulation register at each DUMP.

Alternatively, the epoch counters can be left free–running and the offset can be added by the software each time it reads the epoch registers. Note that if the integration is performed across bit boundaries, the integration result can be very small.

7.4.8 Reading the Measurement Data

At each TIC, the measurement data is latched in the Measurement Data registers (CHx_EPOCH, CHx_CODE_PHASE, CHx_CARRIER_DCO_PHASE, CHx_CARRIER_CYCLE_HIGH, CHx_CARRIER_CYCLE_LOW, and CHx_CODE_DCO_PHASE).

The ACCUM_STATUS_B or MEAS_STATUS_A register must be polled at a rate greater than the TIC rate (to see if a TIC has occurred), otherwise measurement data will be lost. The ACCUM_INT or MEAS_INT events can be used to instigate this operation. The reading of measurement data can be either interrupt driven or polled.

For the interrupt driven method the microprocessor reads the ACCUM_STATUS_B or MEAS_STATUS_A register after each MEAS_INT, and if the TIC bit is set, subsequently reads the Measurement data.

For the polled method, the ACCUM_STATUS_A register is always read following every ACCUM_INT. In addition, the ACCUM_STATUS_B register is read on each ACCUM_INT to ensure no Accumulated Data has been missed and to check the TIC bit (along with several other status bits). The software tests the TIC bit to determine if new Measurement Data is available to be read.

7.4.9 PRESET Mode

Each channel can be programmed into PRESET mode by writing a High into the PRESET/UPDATEB bit of the CHx_SATCNTL register.

When a TIC occurs, the satellite code, epoch value and slew numbers are loaded, and a new phase programmed into the Code DCO regardless of its previous value. Prior to the TIC, the channel operates with its previous settings.

PRESET Mode has no effect on the Carrier DCO and Carrier Cycle Counter.

If PRESET mode is initiated, it should be allowed to operate to completion. The required sequence of operations is as follows:

1) Write into CHx_SATCNTL to select the PRESET mode, together with the appropriate new settings.

2) Load the Code and Carrier DCO increment values. Note: These will take effect immediately thereby influencing the current measurements.

3) Load the following Registers: CHx_CODE_DCO_PHASE, CHx_CODE_SLEW and CHx_EPOCH_COUNT_LOAD. It is important that the CHx_EPOCH_COUNT_LOAD occurs last, because it enables the PRESET operation on the next TIC.

7.4.10 Interrupts

There are two interrupt sources: ACCUM_INT and MEAS_INT. Their sense is dependent upon the selected microprocessor interface mode. The default ACCUM_INT period is 505.05μs. However, it can be reconfigured via the PROG_ACCUM_INT register or by changing the INTERRUPT_PERIOD or FRONT_END_MODE bits in the SYSTEM_SETUP register. The default MEAS_INT period is 50ms. However, this can be reconfigured via the PROG_TIC_HIGH and PROG_TIC_LOW registers. Both ACCUM_INT and MEAS_INT are applied to the Interrupt Controller (INTC) in the Firefly MF1, and can hence be enabled or disabled from there.

7.4.11 Signal Path Delay Introduced by Hardware Signal Processing

When it is desired to generate an accurate time reference from GPS signals or to time–stamp position fixes the delays in the receiver must be allowed for. The signal path delay has two components, an Analogue path delay that varies with temperature and component tolerances; and a Digital path delay that is constant if oscillator drift variations are neglected.

The Digital delay is easier to estimate and is made up of the following:

i. The time from the sampling edge of the SIGN and MAG bits in the front end (SAMPCLK) to the re–sampling in the Sample Latch. This is 175ns, less the propagation delay of SAMPCLK to the Front–end, giving approx. 125ns (including the delay in the series 1.5kΩ resistor in the SAMPCLK feed).

ii. Plus the time for the correlation in the Correlator on these same SIGN and MAG bits (125ns).

iii. Plus the delay in the accumulator to latch the sampled data (175ns).

iv. Less the time between the correlation and the TIC clock phase that is before the accumulator latch phase (75ns).

This gives a total Signal path delay of 400ns, less the SAMPCLK delay.

7: 12-Channel Correlator

The Analogue delay through the RF Front-end of the GPS receiver is set by such parameters as group delay in filters. For the bandwidths used for C/A code will be in the region of 1 to 2 μs. This will normally swamp the digital delay, but this can be measured and corrected for.

7.4.12 Integrated Carrier Phase Measurement

The Correlator tracking channel hardware allows measurement of integrated carrier phase through the CHx_CARRIER_CYCLE_HIGH and _LOW and the CHx_CARRIER_DCO_PHASE registers, which are part of the Measurement Data sampled at every TIC. The CHx_CARRIER_CYCLE_HIGH and _LOW registers contain the 20-bit number of positive–going zero crossings of the Carrier DCO. This will be one more than the number of full cycles elapsed (4-bits are in _HIGH and 16-bits in _LOW register). The CHx_CARRIER_DCO_PHASE register contains the cycle fraction or phase, with 10-bit resolution to give 2π / 1024 radian increments.

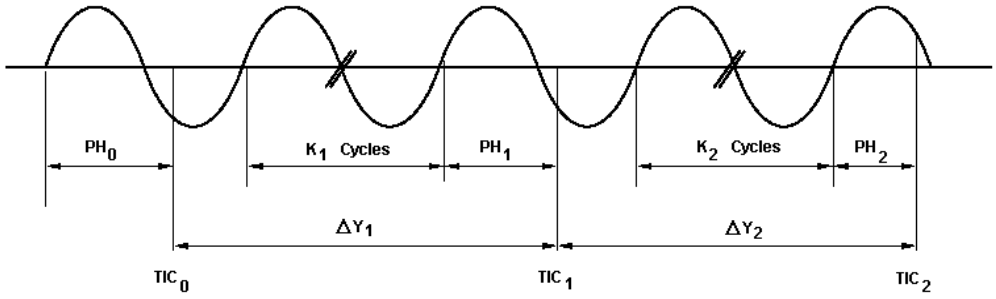
To get the Integrated Carrier Phase over several TIC periods all that is needed is to read the CHx_CARRIER_CYCLE_HIGH and _LOW registers at every TIC and sum the readings. This gives a number one higher than the number of complete carrier cycles, when a carrier cycle is measured from one positive–going zero crossing to the next. To this number, the fractional carrier cycle at the end has to be added, and the fractional carrier cycle at the beginning has to be subtracted. Both numbers are read from the CHx_CARR_DCO_PHASE register. The total phase change can be calculated as follows:

Integrated Carrier Phase =

$$2\pi * \sum \text{No.s in Carrier Cycle Counter} + \text{final Carrier DCO phase} - \text{Initial Carrier DCO phase}$$

Figure 7.4 below shows how this equation is derived.

This Integrated Carrier Phase may be related to the delta–range (the change in distance to each satellite). When used with the orbital parameters of the satellites, the delta–ranges give a measure of the receiver's movement between fixes, which is independent of those fixes and so can be used to smooth them. It can also give a velocity directly. The delta–ranges will be noisy and most of the value is due to satellite movement so the determination of velocity must use data from adequately separated TICs. For position smoothing all delta–ranges may be included in the input to the navigation filter, as that filter will perform a running average of the delta–ranges as well as the ranges.



1. reading at TIC 0 : CHx_CARR_DCO_PHASE 0 = PH 0

2. reading at TIC 1 : CHx_CARR_DCO_PHASE 1 = PH 1
   CHx_CARR_CYCLE 1 = K 1 + 1
3. reading at TIC 2 : CHx_CARR_DCO_PHASE 2 = PH 2
   CHx_CARR_CYCLE 2 = K 2 + 1

$$\Delta Y_1 = 2\pi \times K_1 + (2\pi\ PH_0) + PH_1$$
$$= 2\pi(K_1 + 1)\ PH_0 + PH_1$$
$$= 2\pi \times \left(CHx\_CARR\_CYCLE_1 - \frac{CHx\_CARR\_DCO\_PHASE_0 + CHx\_CARR\_DCO\_PHASE_1}{1024} \right)$$

$$\sum \Delta Y = 2\pi \times \left(\left(\sum_{i=1}^{last} CHx\_CARR\_CYCLE \right) - \frac{CHx\_CARR\_DCO\_PHASE_0 + CHx\_CARR\_DCO\_PHASE_{last}}{1024} \right)$$ Note: The Carrier Cycle Counter value is stored at every TIC and the Counter is reset

Figure 7.4 Integrated carrier phase

7.5 12 Channel Correlator Interface Timing

In addition to the detailed timings associated with individual read and write cycles, the internal architecture of the correlator also imposes limits on cycle to cycle timings (in particular write to write cycle and write to read cycle).

In the GP4020, it must be ensured that no attempts are made to access the 12-channel correlator for the 300ns following the end of a correlator write cycle. However, if the controlling software is to be allowed to write rapidly to the correlator (e.g. block writes), then a more complex bus interface (which inserts wait states) will be required. Note that this limitation only applies after correlator writes, and does not apply to writes to the correlator X_DCO_INCR_HIGH address.

The correlator section of the GP4020 uses a multi–phase clock internally, and the correlator registers load on specific clock phases. At the end of a write cycle, the falling edge of the internal write strobe latches both the relevant address and data bits. This data is then loaded from the internal data bus to the relevant register at some time during the following 300ns. A write cycle to the Correlator with no writes in the preceding 300ns (314ns), may be performed immediately, so long as the detailed signal timings are met. However, subsequent read or write cycles to the Correlator after this write cycle may need to be delayed if they would modify the internal address or data lines. Correlator read cycles with no write cycles in the preceding 300ns (314ns) are self–contained, and do not delay subsequent cycles. An isolated read cycle requires only sufficient wait states to meet the detailed signal timings.

7.5.1 Write Cycle To Read Cycle Timings

As described previously, the internal write cycle of the Correlator takes 300ns. Only once the write cycle is complete will the correlator address decoders switch to decoding the current address. The correlator uses a pre–charged internal data out bus and hence the decoded address lines must be stable before the internal bus drivers are enabled (when the read strobe goes high). Consequently, the read strobe must be held Low until some time after the end of the 300ns (314ns) internal write cycle, to allow sufficient internal address set-up time

7.5.2 Write Cycle To Write Cycle Timings

The internal write cycle of the correlator takes 300ns after the falling edge of the write strobe. During this time the write internal address and data busses (latched by write) must not be modified. If a second write follows the first, the second write cycle must be delayed such that it ends no earlier than 300ns after the end of the previous write. The 'end' being a falling edge on the internal write strobe. The specific interface signal timings must also be met.

Writes to the Correlator register X_DCO_INCR_HIGH need not incur subsequent delays, since writes to this location do not instigate an internal write cycle. A write to this address must always be followed by a write to either a CHX_CARRIER_DCO_INCR_LOW or a CHX_CODE_DCO_INCR_ LOW register. It is this second associated write which instigates the internal write cycle.

Note that the exact number of wait states which need to be inserted after a correlator write is not fixed. If the processor were to perform a correlator write then spend 400ns accessing a different peripheral, subsequent correlator reads and writes would incur no additional delay.

7.6 12-Channel Correlator Register Maps

The register map of the 12-Channel Correlator within the GP4020 is shown in *Table 7.2 below*. The Base Address for the GP4020 12-channel Correlator block is 0x4010 0000. The addresses are complete, and it should be noted that all the register addresses are 32-bit word–aligned but are 16-bits wide, i.e. A0 and A1 are not used. Adjacent register addresses thus increment by four. Data can be written to and read from the correlator in 32-bit width, but the 16MSBs of each 32-bit read or write will be ignored.

Address Offset	Register	Direction	Function
0x000 to 0x01C	CH0 Control	*(see Table 7.3)*	Correlator Channel Control Registers *(see Table 7.3)*
0x020 to 0x03C	CH1 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x040 to 0x05C	CH2 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x060 to 0x07C	CH3 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x080 to 0x09C	CH4 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x0A0 to 0x0BC	CH5 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x0C0 to 0x0DC	CH6 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x0E0 to 0x0FC	CH7 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x100 to 0x11C	CH8 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x120 to 0x13C	CH9 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x140 to 0x15C	CH10 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x160 to 0x05C	CH11 Control	*(see Table 7.3)*	*(see Table 7.3)*
0x180 to 0x19C	MULTI Control	*(see Table 7.3)*	*(see Table 7.3)*
0x1A4	X_DCO_INCR_HIGH	Write	
0x1AC	PROG_ACCUM_INT	Write	ACCUM_INT Period Counter
0x1B4	PROG_TIC_HIGH	Write	Bits [20:16] of TIC Period Counter
0x1BC	PROG_TIC_LOW	Write	Bits [15:0] of TIC Period Counter

Address Offset	Register	Direction	Function
0x1C0 to 0x1DC	ALL Control	*(see Table 7.3)*	*(see Table 7.3)*
0x1EC	TIMEMARK_CONTROL	Write	Configure Raw Timemark output
0x1F0	TEST_CONTROL	Write	Set-up correlator test modes
0x1F4	MULTI_CHANNEL _SELECT	Write	Select channels for "MULTI"
0x1F8	SYSTEM_SETUP	Write	Set-up top-level correlator configs
0x1FC	RESET_CONTROL	Write	Channel Reset (Disable)
0x200 to 0x20C	Status Registers	*(see Table 7.5)*	*(see Table 7.5)*
0x210 to 0x21C	CH0 Accumulate	*(see Table 7.4)*	Correlator Channel Data Accumulation Registers *(see Table 7.4)*
0x220 to 0x22C	CH1 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x230 to 0x23C	CH2 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x240 to 0x24C	CH3 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x260 to 0x26C	CH5 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x270 to 0x27C	CH6 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x280 to 0x28C	CH7 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x290 to 0x29C	CH8 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x2A0 to 0x2AC	CH9 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x2B0 to 0x2BC	CH10 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x2C0 to 0x2BC	CH11 Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x2D0 to 0x2DC	MULTI Accumulate	*(see Table 7.4)*	*(see Table 7.4)*
0x2E0 to 0x2EC	ALL Accumulate	*(see Table 7.4)*	*(see Table 7.4)*

Table 7.2 12-channel correlator (CORR) Register Map

Addresses for each of the Correlator Registers may be calculated from a base address with an increment for a particular register.

In both the Accumulate and Control register sections in *Table 7.2 above*, there are some addresses labelled ALL or MULTI in place of CHx. By writing to these addresses, either all the channels or a selection of channels set by MULTI_CHANNEL_SELECT will be written to in one operation. This facility may be used to initialise the system quickly or to load the next search settings with little bus use. This is a write only function and the corresponding CHx read functions are not available at addresses labelled ALL or MULTI.

7.6.1 Tracking Channel Control Registers

Each Tracking channel has the Control registers as shown in *Table 7.3 below*. Each address has an independent read and write function. Complete address offset for each Channel Control register can be determined using:

Correlator Register Address Offset = CHx_Control Base Address + Control Register Offset

For Example, CH3_CODE_DCO_INCR_LOW = 0x060 + 0x018 = 0x078

It can be seen in *Table 7.3 below* that the addresses for the Channel Control registers are used to Control the channel in write mode, but give the channel Measurement Data when in read mode.

7.6.2 Tracking Channel Data Accumulation Registers

Each Tracking channel has the Data Accumulation registers as shown in *Table 7.4 on page 67*. Each address has an independent read and write function. Complete address offset for each Channel Control register can be determined using:

Correlator Register Address Offset =
CHx_Accumulate Base Address + Accumulate Register Offset

For Example, CH3_Q_TRACK = 0x240 + 0x004 = **0x244**

Address Offset	Register	Direction	Function
CHx_ Control + 0x00	CODE_SLEW	READ	11-bit Code Slew value
	SATCNTL	WRITE	Configure C/A Code generator
+ 0x04	CODE_PHASE	READ	11-bit Code Phase Count
	CODE PHASE COUNTER[1]	WRITE	Load Code Phase Counter *(test mode only)*
+ 0x08	CARRIER_CYCLE_LOW	READ	16 LSBs of Carrier Cycle Count
	CARRIER_CYCLE_ COUNTER[1]	WRITE	Load Carrier Cycle Counter *(test mode only)*
+ 0x0C	CARRIER_DCO_PHASE	READ	10 MSBs of Carrier Phase Accumulator, sampled at TIC
	CARRIER_DCO_INCR_ HIGH	WRITE	10 MSBs of Carrier DCO phase increment
+ 0x10	EPOCH (Latched)	READ	1ms and 20ms EPOCH Counter values latched at last TIC event.
	CARRIER_DCO_INCR_ LOW	WRITE	16 LSBs of Carrier DCO phase increment
+ 0x14	CODE_DCO_PHASE	READ	10 MSBs of Code Phase Accumulator, sampled at TIC
	CODE_DCO_INCR_HIGH	WRITE	9 MSBs of Code DCO phase increment
+ 0x18	CARRIER_CYCLE_HIGH	READ	4 MSBs of Carrier Cycle Count
	CODE_DCO_INCR_LOW	WRITE	16 LSBs of Code DCO phase increment
+ 0x1C	EPOCH_CHECK (Not latched)	READ	Instantaneous values of 1ms and 20ms EPOCH counters.
	EPOCH_COUNT_LOAD	WRITE	1ms and 20ms EPOCH Counter load values.

Table 7.3 CORR Tracking Channel Control Registers Map

Note 1: The CODE_PHASE_COUNTER and CARRIER_CYCLE_CONTROL registers can only be written to if 'Test' mode has been selected by setting bit 3 of the TEST CONTROL register to High.

Address Offset		Register	Direction	Function
CHx_ Accumulate	+ 0x00	I_TRACK	READ	Integrate and Dump Values for I tracking arm in correlator channel X.
		CODE_SLEW _COUNTER	WRITE	Sets number of code half-chips to slew the C/A code generator at next DUMP event.
	+ 0x04	Q_TRACK	READ	Integrate and Dump Values for Q tracking arm in correlator channel X.
		ACCUM_RESET	WRITE	Reset ACCUM_STATUS_X registers.
	+ 0x08	I_PROMPT	READ	Integrate and Dump Values for I prompt arm in correlator channel X.
		not used	WRITE	
	+ 0x0C	Q_PROMPT	READ	Integrate and Dump Values for Q prompt arm in correlator channel X.
		CODE_DCO _PRESET_ PHASE	WRITE	8 MSBs of CODE_DCO phase to be loaded at next TIC event, in PRESET mode.

Table 7.4 CORR Tracking Channel Data Accumulation Registers Map

Address Offset	Register	Direction	Function
0x200	ACCUM_STATUS_C	READ	Indicates either "Early" or "Late" codes, on each correlator channel.
	STATUS	WRITE	Latches data on ALL ACCUM_STATUS registers
0x204	MEAS_STATUS_A	READ	Indicates if Measurement data has been missed, on each correlator channel.
	not used	WRITE	
0x208	ACCUM_STATUS_A	READ	Indicates if new Accumulation data is available on each correlator channel.
	not used	WRITE	
0x20C	ACCUM_STATUS_B	READ	Indicates if new Accumulation data has been missed, on each correlator channel.
	not used	WRITE	

Table 7.5 CORR Tracking Channel Status Registers Map

Apart from the Code and Carrier DCO increment values, all data transfers are only 16-bits wide. Writes to the Code and Carrier DCO's are 32-bit data transfers. The _HIGH word should be written first and will be retained in the 16- to 32-bit interface until the _LOW word is written. The _LOW word write must occur as the next write to the chip. All 32-bits will then be transferred into the DCO increment register. Data is written to an input buffer in the 16- to 32-bit interface and will be transferred to its destination register during the next full cycle of the 7 (or 6) phase clock. Write cycles should therefore have a period of at least 300ns. The X_DCO_INCR_HIGH may be used to write the high bits of the increment number to any or all DCO's as an alternative to using the CHx_CODE / CARRIER_DCO_INCR– _HIGH addresses. By using this address, there is no need to wait 300ns before writing the _LOW part. For further information, refer to *Section 7.5 "12 Channel Correlator Interface Timing" on page 63*.

The bit assignments for the Correlator registers are given below, but two write–only registers do not have any data bits, these are:

1) A write to the CHx_ACCUM_RESET register (irrespective of what data is written) will reset the ACCUM_STATUS_A, ACCUM_STATUS_B, and ACCUM_STATUS_C registers for that channel.

2) A write to the STATUS register (irrespective of what data is written) will latch the state of the various status flags into ACCUM_STATUS_A, ACCUM_STATUS_B, ACCUM_STATUS_C registers for all channels. This allows polling based, rather than Interrupt driven tracking scheme.

The registers are listed in alphabetical order and not in address order to allow easy reference to each section. Unless otherwise stated the LSB is bit 0 and the MSB is bit 15 or as far up the register as there is data. Note that most registers do not have both read and write functions, and many addresses are shared between read–only and write–only registers having different functions.

7.6.3 ACCUM_STATUS_A Register - Read Address offset 0x208

ACCUM_STATUS_A is a register containing the state of twelve status bits sampled and latched on the active edge of every ACCUM_INT. They can also be sampled and latched on request, by performing a write operation to STATUS. (This is safe only if the interrupts are stopped, by setting INTERRUPT_ENABLE bit to LOW in the SYSTEM_SETUP register.) The microprocessor must respond to each ACCUM_INT and read the channel registers before the next DUMP is due in that channel.

Bit No.	Mnemonic	Description	Reset Value	R/W
15	ACCUM_INT	Set HIGH at each occurrence of ACCUM_INT; Reset by reading ACCUM_STATUS_A register. This status bit is reset by a hardware master reset but not by a software reset (MRB).	0	R
14	*Not used*	'0' when read	0	R
13	*Not used*	'0' when read	0	R
12	*Not used*	'0' when read	0	R
11	CH11_NEW_ACCUM_DATA	'1' = a DUMP has occurred in Correlator Channel 11, and that new Accumulated Data is available to be read. '0' = no new data available. Each bit is cleared by the trailing edge of a read of the associated CHx_Q_PROMPT register or by a write to CHx_ACCUM_RESET.	0	R
10	CH10_NEW_ACCUM_DATA	(as bit 11 but for channel 10)	0	R
9	CH9_NEW_ACCUM_DATA	(as bit 11 but for channel 9)	0	R
8	CH8_NEW_ACCUM_DATA	(as bit 11 but for channel 8)	0	R
7	CH7_NEW_ACCUM_DATA	(as bit 11 but for channel 7)	0	R
6	CH6_NEW_ACCUM_DATA	(as bit 11 but for channel 6)	0	R
5	CH5_NEW_ACCUM_DATA	(as bit 11 but for channel 5)	0	R
4	CH4_NEW_ACCUM_DATA	(as bit 11 but for channel 4)	0	R
3	CH3_NEW_ACCUM_DATA	(as bit 11 but for channel 3)	0	R
2	CH2_NEW_ACCUM_DATA	(as bit 11 but for channel 2)	0	R
1	CH1_NEW_ACCUM_DATA	(as bit 11 but for channel 1)	0	R
0	CH0_NEW_ACCUM_DATA	(as bit 11 but for channel 0)	0	R

Table 7.6 CORR ACCUM_STATUS_A Register

Note that the channel specific bits of this register will not show their new value until after an active edge of ACCUM_INT or a write to the STATUS register. Disabling a channel will however, clear the bit immediately.

7.6.4 ACCUM_STATUS_B Register - Read Address offset 0x20C

ACCUM_STATUS_B is a register containing the state of twelve status bits sampled and latched on the active edge of every ACCUM_INT (as for ACCUM_STATUS_A). They can also be sampled and latched on request, by performing a write operation to STATUS.

Bit No	Mnemonic	Description	Reset Value	R/W
15	DISCIP1_GLITCH	The DISCIP_GLITCH bit will be set HIGH if a glitch to LOW has occurred on the DISCIP pin since the last read of this register. It is cleared by reading this ACCUM_STATUS_B register. This bit is reset by a hardware master reset (NRESET at Low) but not by a software reset. The minimum reliably detectable glitch width is 25ns.	0	R

Bit No	Mnemonic	Description	Reset Value	R/W
14	DISCIP1	The DISCIP1 bit indicates the level on the DISCIP input pin at the time this read occurs. It may be used to interface a hardware condition (such as a ready flag from a UART, or the PLL LOCK signal from a front–end) to the microprocessor without using an interrupt. This bit is not reset by a hardware master reset nor by an MRB.	0	R
13	TIC	Set HIGH at every occurrence of TIC and is cleared by reading this ACCUM_STATUS_B register. This bit can be used as a flag to the microprocessor, to time software module swapping. It is reset by a hardware master reset (NRESET at Low) but not by an MRB in RESET_CONTROL.	0	R
12	MEAS_INT	Provided that interrupts are enabled, the MEAS_INT bit is set High at each TIC and 50 ms before each TIC (if the TIC period is greater then 50 ms), and is cleared by reading this register. This bit can be used to tell the microprocessor that new Measurement Data is available. It is reset by a hardware master reset (NRESET at Low), but not by a software reset.	0	R
11	CH11_MISSED_ ACCUM	'1' = missed Accumulated Data due to a new DUMP in CHx before the previous data has been read. '0' = no missed data. This bit is latched until the associated CHx_ACCUM_RESET is written to. The data is sampled and latched on the active edge of every ACCUM_INT signal and on request by performing a write operation to STATUS (as with ACCUM_STATUS_A).	0	R
10	CH10_MISSED_ ACCUM	(as bit 11 but for channel 10)	0	R
9	CH9_MISSED_ACCUM	(as bit 11 but for channel 9)	0	R
8	CH8_MISSED_ACCUM	(as bit 11 but for channel 8)	0	R
7	CH7_MISSED_ACCUM	(as bit 11 but for channel 7)	0	R
6	CH6_MISSED_ACCUM	(as bit 11 but for channel 6)	0	R
5	CH5_MISSED_ACCUM	(as bit 11 but for channel 5)	0	R
4	CH4_MISSED_ACCUM	(as bit 11 but for channel 4)	0	R
3	CH3_MISSED_ACCUM	(as bit 11 but for channel 3)	0	R
2	CH2_MISSED_ACCUM	(as bit 11 but for channel 2)	0	R
1	CH1_MISSED_ACCUM	(as bit 11 but for channel 1)	0	R
0	CH0_MISSED_ACCUM	(as bit 11 but for channel 0)	0	R

Table 7.7 CORR ACCUM_STATUS_B Register

Note. If any accumulation data is missed due to the reading process being too slow this must be allowed for in the software, such as by checking the Navigation Message data bit transitions independently of the sets of Accumulated Data reads. If too much data is lost the system signal-to-noise ratio will be degraded. The primary purpose of these bits is as a check on how well the tracking routines are working – once the whole design is complete, these bits should not become set.

Channel specific bits of this register will not show their new value until after an active edge of ACCUM_INT or a write to the STATUS register. Disabling a channel will however, clear the bit immediately.

7.6.5 ACCUM_STATUS_C Register - Read Address offset 0x200

ACCUM_STATUS_C is a register containing the state of twelve status bits sampled and latched on the active edge of every ACCUM_INT (as for ACCUM_STATUS_A). They can also be sampled and latched on request, by performing a write operation to STATUS.

Bit No.	Mnemonic	Description	Reset Value	R/W
15	*Not used*	'0' when read	0	R
14	*Not used*	'0' when read	0	R
13	*Not used*	'0' when read	0	R
12	*Not used*	'0' when read	0	R
11	CH11_EARLY_LATEB	Status bit which indicates the code type for the accumulated Data on the Tracking arm of channel 11, when that channel is in Dithering mode. '1' = EARLY code '0' = LATE code. Each individual bit is determined on the DUMP that sets CHx_NEW_ACCUM_DATA to High for that channel.	0	R
10	CH10_EARLY_LATEB	(as bit 11 but for channel 10)	0	R
9	CH9_EARLY_LATEB	(as bit 11 but for channel 9)	0	R
8	CH8_EARLY_LATEB	(as bit 11 but for channel 8)	0	R
7	CH7_EARLY_LATEB	(as bit 11 but for channel 7)	0	R
6	CH6_EARLY_LATEB	(as bit 11 but for channel 6)	0	R
5	CH5_EARLY_LATEB	(as bit 11 but for channel 5)	0	R
4	CH4_EARLY_LATEB	(as bit 11 but for channel 4)	0	R
3	CH3_EARLY_LATEB	(as bit 11 but for channel 3)	0	R
2	CH2_EARLY_LATEB	(as bit 11 but for channel 2)	0	R
1	CH1_EARLY_LATEB	(as bit 11 but for channel 1)	0	R
0	CH0_EARLY_LATEB	(as bit 11 but for channel 0)	0	R

Table 7.8 CORR ACCUM_STATUS_C Register

Note that the channel specific bits of this register will not show their new value until after an active edge of ACCUM_INT or a write to the STATUS register. Disabling a channel will however, clear the bit immediately.

7.6.6 CHx_ACCUM_RESET Register - Offset <CHx_Accumulate> + 0x04

Accumulator Status Register reset. A write of any value to this register will reset all of the status bits in ACCUM_STATUS_A, ACCUM_STATUS_B, and ACCUM_STATUS_C associated with a given channel or all channels. When these locations are written to, the data is irrelevant.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	*Not used*	Reset Accumulator Status Registers	0x0000	W

Table 7.9 CORR CHx_ACCUM_RESET Register

7.6.7 CHx_CARRIER_CYCLE_COUNTER Register - Offset <CHx_Control> + 0x08
MULTI_CARRIER_CYCLE_COUNTER Register - Offset (0x180 + 0x08)
ALL _CARRIER_CYCLE_COUNTER Register - Offset (0x1C0 + 0x08)

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	*Not used*	Write–only in Test-mode only: Value loaded into lower 16-bits of CHx_CARRIER_CYCLE_COUNTER along with zeros into the upper 4-bits. *((A write to these registers only has effect when in test mode (bit 3 of TEST_CONTROL set High)).*	0x0000	W

Table 7.10 CORR CHx_CARRIER_CYCLE_COUNTER Register

7.6.8 CHx_CARRIER_CYCLE_HIGH Register - Offset <CHx_Control> + 0x18

The Correlator tracking channel hardware allows for measurement of integrated carrier phase through the CHx_CARRIER_CYCLE_HIGH and _LOW and the CHx_CARRIER_DCO_PHASE registers, which are part of the Measurement Data sampled at every TIC. The CHx_CARRIER_CYCLE_HIGH and _LOW registers contain the 20-bit number of positive going zero crossings of the Carrier DCO (4-bits are in _HIGH and 16-bits in _LOW). The cycle fraction can be read from the CHx_CARRIER_DCO_PHASE register.

In the CHx_CARRIER_CYCLE counter, a TIC generates two consecutive actions. First it latches the four more significant bits of the cycle counter into CHx_CARRIER_CYCLE_HIGH and the 16 less significant bits into CHx_CARRIER_CYCLE_LOW. Then it resets the cycle counter.

After each TIC, every time the Carrier DCO accumulator generates an overflow because of a carrier cycle being completed, the cycle counter increments by one.

The nominal CARRIER DCO frequency with no Doppler and no oscillator drift compensation is 1.405396825 MHz, so in 100 ms, there will be about 140540 cycles.

In almost all applications, the number of Carrier DCO cycles does not vary much from one TIC interval to another. It is possible to predict the Most Significant Bits of the value, and then only read the CHx_CARRIER_CYCLE_LOW register.

CHx_CARRIER_CYCLE_HIGH and _LOW contents are not protected by an overwrite protection mechanism and so must be read before the next TIC. For further information on the Carrier Cycle Counter, refer to *Section 7.4 "Controlling the 12 Channel Correlator" on page 59.*

Bit No.	Mnemonic	Description	Reset Value	R/W
15:4	*Not used*	'0' when read	0	R
3:0	CHx_CARRIER_CYCLE [19:16]	Bits 19:16 of the 20-bit Carrier Cycle Count	0000	R

Table 7.11 CORR CHx_CARRIER_CYCLE_HIGH Register

7.6.9 CHx_CARRIER_CYCLE_LOW Register - Offset <CHx_Control> + 0x08

The CHx_CARRIER_CYCLE_HIGH and LOW registers contain the 20-bit number of positive going zero crossings of the Carrier DCO (4-bits are in HIGH and 16-bits in LOW). Refer to CHx_CARRIER_CYCLE_HIGH for more information

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	CHx_CARRIER_CYCLE [15:0]	Bits 15:0 of the 20-bit Carrier Cycle Count	0x0000	R

Table 7.12 CORR CHx_CARRIER_CYCLE_LOW Register

7.6.10 CHx_CARRIER_DCO_INCR_HIGH Register - Offset <CHx_Control> + 0x0C
MULTI_CARRIER_DCO_INCR_HIGH Register - Offset 0x180 + 0x0C
ALL_CARRIER_DCO_INCR_HIGH Register - Offset 0x1C0 + 0x0C

The _CARRIER_DCO_INCR_HIGH Register contains the 10 Most Significant bits of a 26-bit value used to set the frequency of the Carrier DCO in the correlator channel selected. The programmed value is treated as an increment of a Minimum frequency step.

The contents of registers _CARRIER_DCO_INCR_HIGH and _CARRIER_DCO_INCR_LOW are combined to form the 26-bits of the CHx_CARRIER_DCO_INCR register, the carrier DCO phase increment number. In order to write successfully, the top 10-bits must be written first, to any of the _HIGH addresses. They will be stored in a buffer and only be transferred into the increment register of the DCO together with the _LOW word.

A 26-bit increment number is adequate for a 27-bit accumulator DCO, as the increment to the MSB is always zero. The LSB of the INCR register represents a step given by:

Min Step Frequency = (40MHz / 7) / 2^{27} = 42.57475mHz

Output Frequency = CHx_CARRIER_DCO_INCR * Min Step Frequency.

With a GP2015/GP2010 style front end, the nominal value of the IF is 1.405396826 MHz before allowing for Doppler shift or crystal error. Writing 0x01F7 B1B9 into the CHx_CARRIER_DCO_INCR register will generate a local oscillator frequency of 1.405396845 MHz.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:10	*Not used*		-	W
9:0	CHx_CARRIER_DCO_INCR [25:16]	Bits 25:16 of the 26-bit Carrier DCO Increment Register. Must be written before CHx_CARRIER_DCO_INCR_LOW values.	0x000	W

Table 7.13 CORR CHx_CARRIER_DCO_INCR_HIGH Register

7.6.11 CHx_CARRIER_DCO_INCR_LOW Register - Offset <CHx_Control> + 0x10
MULTI_CARRIER_DCO_INCR_LOW Register - Offset 0x180 + 0x10
ALL_CARRIER_DCO_INCR_LOW Register - Offset 0x1C0 + 0x10

This register contains the 16 least significant bits for the CHx_CARRIER_DCO_INCR register. Refer to "CHx_CARRIER_DCO_INCR_HIGH" for more information.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	CHx_CARRIER_DCO_INCR [15:0]	Bits 15:0 of the 26-bit Carrier DCO Increment Register	0x0000	W

Table 7.14 CORR CHx_CARRIER_DCO_INCR_LOW Register

7.6.12 CHx_CARRIER_DCO_PHASE - Read Address Offset <CHx_Control> + 0x0C

This register contains the 10-bits of the Carrier DCO phase accumulator, and indicates the phase of a carrier DCO cycle, as a 10-bit sub-multiple of one DCO carrier cycle, as sampled at the last TIC. The weight of the least significant bit is 2π / 1024 radians of a Carrier DCO cycle. These bits form an unsigned integer valid from 0 to 1023. CHx_CARRIER_DCO_PHASE provides sub-cycle phase-measurement information and so complements the information given by CHx_CARRIER_CYCLE_HIGH and _LOW.

The register value is latched on each TIC and is not protected by any overwrite protection mechanism.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:10	*Not used*	'0' when read	0	R
9:0	CHx_CARRIER_DCO_PHASE [9:0]	Bits 9:0 of the 10-bit Carrier DCO Phase Count.	0x000	R

Table 7.15 CORR CHx_CARRIER_DCO_PHASE Register

7.6.13 CHx_CODE_DCO_INCR_HIGH Register - Offset <CHx_Control> + 0x14
MULTI_CODE_DCO_INCR_HIGH Register - Offset 0x180 + 0x14
ALL_CODE_DCO_INCR_HIGH Register - Offset 0x1C0 + 0x14

The _CODE_DCO_INCR_HIGH Register contains the nine Most Significant bits of a 25-bit value used to set the frequency of the Code DCO in the correlator channel selected. The programmed value is treated as an increment of a Minimum frequency step.

The contents of registers _INCR_HIGH and _INCR_LOW are combined to form the 25-bits of the CHx_CODE_DCO_INCR register, the Code DCO phase increment number. In order to write successfully, the top 9-bits must be written first, to any of the _HIGH addresses. They will be stored in a buffer and only be transferred into the increment register of the DCO together with the _LOW word. A 25-bit increment number is adequate for a 26-bit accumulator DCO as the increment to the MSB is always zero.

The LSB of the INCR register represents a step given by:

Min Step Frequency = (40MHz / 7)/2^{26} = 85.14949mHz

Output Frequency = CHx_CODE_DCO_INCR * Min Step Frequency.

Note: The Code DCO drives the Code Generator to give half chip time steps and so must be programmed to twice the required chip rate. This means that the chip rate resolution is 42·57475mHz. The nominal frequency is 1.023000000 MHz before allowing for Doppler shift or crystal error. Writing 0x016E A4A8 into the CHx_CODE_DCO_INCR register will generate a chip rate of 1.022999968 MHz.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:9	*Not used*		-	W
8:0	CHx_CODE_DCO_INCR [24:16]	Bits 24:16 of the 25-bit Carrier DCO Increment Register. Must be written before CHx_CODE_DCO_INCR_LOW values.	0x000	W

Table 7.16 CORR CHx_CODE_DCO_INCR_HIGH Register

7.6.14 CHx_CODE_DCO_INCR_LOW Register - Offset <CHx_Control> + 0x18
MULTI_CODE_DCO_INCR_LOW Register - Offset 0x180 + 0x18
ALL_CODE_DCO_INCR_LOW Register - Offset 0x1C0 + 0x18

This register contains the 16 least significant bits for the CHx_CARRIER_DCO_INCR register. Refer to "CHx_CODE_DCO_INCR_HIGH" for more information.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	CHx_CODE_DCO_INCR [15:0]	Bits 15:0 of the 25-bit Code DCO Increment Register	0x0000	W

Table 7.17 CORR CHx_CODE_DCO_INCR_LOW Register

7.6.15 CHx_CODE_DCO_PHASE Register - Offset <CHx_Control> + 0x14

This register contains the 10-bits of the Code DCO phase accumulator, and indicates the phase of a Code DCO cycle, as a 10-bit sub-multiple of one Code DCO cycle, as sampled at the last TIC. The weight of the least significant bit is 2π / 1024 radians of a Code DCO cycle, 2π being half of a code chip. Therefore, the pseudorange resolution is 1/2048 of a chip, (equivalent to 0.15 metre or 0.5ns). These bits form an unsigned integer valid from 0 to 1023. CHx_CODE_DCO_PHASE provides sub-cycle phase-measurement information and so complements the information given by CHx_CODE_CYCLE_HIGH and _LOW.

The register value is latched on each TIC and is not protected by any overwrite protection mechanism.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:10	*Not used*	'0' when read.	-	R
9:0	CHx_CODE_DCO_PHASE [24:16]	Bits 9:0 of the 10-bit Code DCO Phase Count.	0x000	R

Table 7.18 CORR CHx_CODE_DCO_PHASE Register

7.6.16 CHx_CODE_DCO_PRESET_PHASE Register - Offset <CHx_Accum.> + 0x0C
MULTI_CODE_DCO_PRESET_PHASE Register - Offset 0x2D0 + 0x0C
ALL_CODE_DCO_PRESET_PHASE Register - Offset 0x2E0 + 0x0C

In PRESET mode, the 8-bits of the CHx_CODE_DCO_PRESET_PHASE register, with zeros filling the lower bits, are transferred to the CODE DCO accumulator on the next TIC. The previous accumulator phase is totally overwritten. The PRESET_PHASE register is a write–only register and it can be written to at any time in PRESET mode or in UPDATE mode, but only has effect when PRESET mode is entered.

The weight of the least significant bit of PRESET phase is 2π / 256 radians of a half chip cycle. In UPDATE mode, this register has no use other than as preparation for PRESET mode.

Refer to *Section 7.4.9 "PRESET Mode" on page 61*, for further information on PRESET mode.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:8	*Not used*	.	-	W
7:0	CHx_CODE_DCO_PRESET_ PHASE [24:16]	More significant bits (25 to 18) of the Code DCO phase which is to be loaded at the next TIC event in PRESET mode.	0x00	W

Table 7.19 CORR CHx_CODE_DCO_PRESET_PHASE Register

7.6.17 CHx_CODE_PHASE Register - Read Offset <CHx_Control> + 0x04
CHx_CODE_PHASE_COUNTER Register - Write Offset <CHx_Control> + 0x04
MULTI_CODE_PHASE_COUNTER Register - Write Offset 0x180 + 0x04
ALL_CODE_PHASE_COUNTER Register - Write Offset 0x1C0 + 0x04

This register is primarily a Read Register (i.e. CHx_CODE_PHASE). However, if 'Test' mode has been selected by setting TM_TEST (TEST_CONTROL[3]) to '1', the CHx_CODE_PHASE_COUNTER registers can be written to. This is a test-mode, and is hence not normally required.

A read of CHx_CODE_PHASE[10:0] indicates the state of the Code Phase Counter, an 11-bit binary up-counter, clocked by the Code generator clock. The Phase is expressed as a number of half code chips and ranges from 0 to 2046 chips. A reading of 2046 is very rare and can only occur if the TIC captures the Code phase just after the counter reaches 2046 and before a DUMP from the C/A Code Generator resets it. DUMP also increments the Epoch counter, so the meaning of a phase value of 2046 + the previous Epoch value is the same as a phase value of (0 + the incremented Epoch value), and either is valid. If a TIC occurs during a Code Slew, the reading will be '0' and that channel's Measurement Data is of no use.

A write to CHx_CODE_PHASE_COUNTER[10:0] loads the written 11-bit value to Code Phase Counter for the channel concerned. This is a Test Mode only, and is NOT required for normal use.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:11	*Not used*	'0' when read.	-	R
10:0	CHx_CODE_PHASE [10:0]	Bits 10:0 of the 11-bit Code Phase Count.	0x000	R

Table 7.20 CORR CHx_CODE_PHASE Register

Bit No.	Mnemonic	Description	Reset Value	R/W
15:11	*Not used*		-	W
10:0	CHx_CODE_PHASE _COUNTER[10:0]	Bits 10:0 of the 11-bit Code Phase Count. Write to the space only possible if TEST_CONTROL[3] ('TM_TEST') set to '1'.	0x000	W

Table 7.21 CORR CHx_CODE_PHASE_COUNTER Register

7.6.18 CHx_CODE_SLEW Register - Read Address Offset <CHx_Control> + 0x00
CHx_CODE_SLEW_COUNTER Register
- Write Address Offset <CHx_Accumulate> + 0x00
MULTI_CODE_SLEW_COUNTER Register - Write Address Offset 0x2D0 + 0x00
ALL_CODE_SLEW_COUNTER Register - Write Address Offset 0x2E0 + 0x00

This register space is primarily a Write Register (i.e. CHx_CODE_SLEW_COUNTER). However, the register can also be read for test purposes, but does not have any system use.

A write to CHx_CODE_SLEW_COUNTER[10:0] gives an unsigned integer ranging from 0 to 2047. This represents the number of code half chips to be slewed immediately after the next DUMP if in UPDATE mode or after the next TIC, if in PRESET mode. Since there are only 2046 half chips in a GPS C/A code, a programmed value of 2047 is equivalent to a programmed value of 1, but the next DUMP event will take place 1 ms later. In PRESET mode, the slew timing is set only by TIC, which will also reset the code generator (no DUMP is needed). A non-zero slew must always be programmed when using PRESET mode.

The CHx_CODE_SLEW register can be written to at any time. If two accesses have taken place before a DUMP in UPDATE mode or before a TIC when in PRESET mode, the latest value will be used at the next slew operation. During the time a slew process is being executed, any further write access to the CHx_CODE_SLEW register will be stored until the following DUMP and then cause the transfer of this new value into the counter. This situation may be avoided by synchronising the access with the associated CHx_NEW_ACCUM_DATA status bit.

If a channel is inactive, a non-zero slew value should be written into CHx_CODE_SLEW before the channel is released. This write will be acted on immediately the reset is released.

If a TIC occurs during or soon after a slew, the channel will not be locked to the satellite, so the Measurement Data for that channel will not be of use.

The ability to read the Slew counter is included only for testing hardware or software and has no other use. It will only give a non-zero result if the read occurs during the actual slew operation. An example of a slewing event is shown in *Figure 7.5 below.*

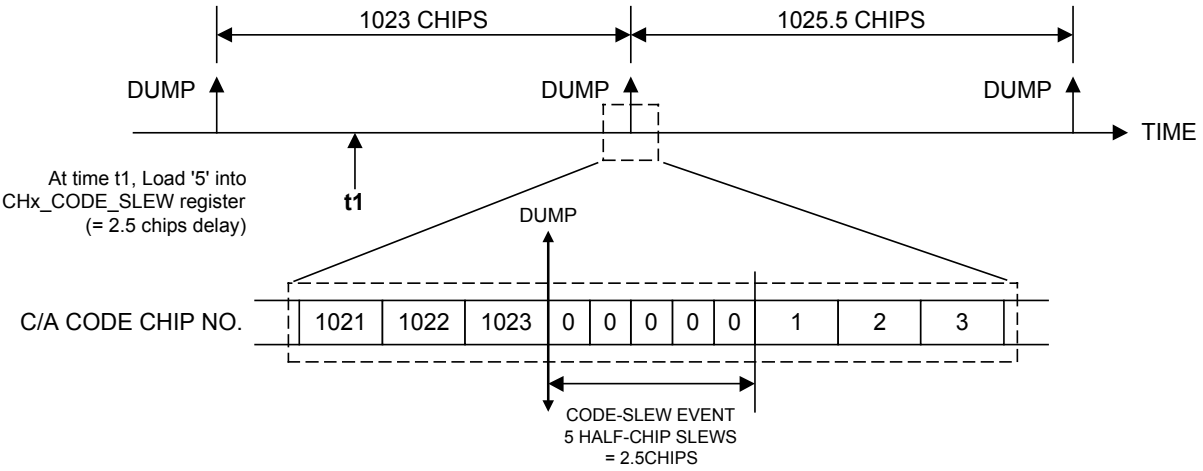


Figure 7.5 Slew timing in UPDATE Mode

Bit No.	Mnemonic	Description	Reset Value	R/W
15:11	*Not used*		-	W
10:0	CHx_CODE_SLEW _COUNTER [10:0]	Bits 10:0 of the 11-bit Code Slew Count, in steps of half a chip.	0x000	W

Table 7.22 CORR CHx_CODE_SLEW_COUNTER Register

Bit No.	Mnemonic	Description	Reset Value	R/W
15:11	*Not used*	'0' when read.	-	R
10:0	CHx_CODE_SLEW[10:0]	Test register only. Indicates a non-zero result if read whilst actual slew occurs.	0x000	R

Table 7.23 CORR CHx_CODE_SLEW Register

7.6.19 CHx_EPOCH_CHECK Register - Read Address Offset <CHx_Control> + 0x1C

This register address gives the instantaneous value of the CHx_1MS_EPOCH and the CHx_20MS_EPOCH counters. It can be used to verify if the software has properly initialised the Epoch counters. Its value is not latched and is incremented on each DUMP. To ensure the correct result, this register should be read only when there is no possibility of getting a DUMP during the read cycle, by synchronising the read to NEW_ACCUM_DATA.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:14	*Not used*	'0' when read.	-	R
13:8	CHx_20MS_EPOCH[5:0]	Instantaneous value of the CHx_20MS_EPOCH. Valid range = 0 to 49	0x00	R
7:5	*Not used*	'0' when read.	-	R
4:0	CHx_1MS_EPOCH[4:0]	Instantaneous value of the CHx_1MS_EPOCH Valid range = 0 to 19	0x00	R

Table 7.24 CORR CHx_EPOCH_CHECK Register

7.6.20 CHx_EPOCH Register - Read Address Offset <CHx_Control> + 0x10

This register gives the values of the CHx_1MS_EPOCH and the CHx_20MS_EPOCH counters, which were latched at the last TIC event.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:14	*Not used*	'0' when read.	-	R
13:8	CHx_20MS_EPOCH[5:0]	Value of the CHx_20MS_EPOCH counter, sampled at the last TIC event. Valid range = 0 to 49	0x00	R
7:5	*Not used*	'0' when read.	-	R
4:0	CHx_1MS_EPOCH[4:0]	Value of the CHx_1MS_EPOCH counter, sampled at the last TIC event. Valid range = 0 to 19	0x00	R

Table 7.25 CORR CHx_EPOCH Register

7.6.21 CHx_EPOCH_COUNT_LOAD Register - Write Offset <CHx_Control> + 0x1C
MULTI_EPOCH_COUNT_LOAD Register - Write Address Offset 0x180 + 0x1C
ALL_EPOCH_COUNT_LOAD Register - Write Address Offset 0x1C0 + 0x1C

This register is used to load the EPOCH counters with values that can synchronise them with the GPS Data message from a GPS satellite. The 20ms EPOCH Counter together with the 1ms EPOCH Counter covers a range from 0ms to 999ms. The execution of the transfer of data from this register to the EPOCH counters is determined by the current channel mode: PRESET or UPDATE.

In UPDATE mode, the data written into these registers is immediately transferred to the 1 ms and 20 ms epoch counters.

In PRESET mode however, the data is transferred only after the next TIC. It is important to load the CHx_EPOCH register last in the PRESET mode loading sequence because the trailing edge of a write to this register enables the whole PRESET operation on the next TIC. Refer to the description of PRESET mode in *Section 7.4.9 on page 61.*

Bit No.	Mnemonic	Description	Reset Value	R/W
15:14	*Not used*		-	W
13:8	CHx_20MS_EPOCH[5:0]	Value to be loaded into the CHx_20MS_EPOCH counter. Valid range = 0 to 49	0x00	W
7:5	*Not used*		-	W
4:0	CHx_1MS_EPOCH[4:0]	Value to be loaded into the CHx_1MS_EPOCH counter. Valid range = 0 to 19	0x00	W

Table 7.26 CORR CHx_EPOCH_COUNT_LOAD Register

7.6.22 CHx_I_TRACK Register - Read Address Offset <CHx_Accumulate> + 0x00
CHx_Q_TRACK Register - Read Address Offset <CHx_Accumulate> + 0x04
CHx_I_PROMPT Register - Read Address Offset <CHx_Accumulate> + 0x08
CHx_Q_PROMPT Register - Read Address Offset <CHx_Accumulate> + 0x0C

These registers hold the Accumulated Data values which result from Code mixing, which are used on each DUMP to store the 16–bit Integrate–and–Dump accumulator results. The values contained in the registers are 2's complement values with the valid range of the data from -2^{15} to $+(2^{15}-1)$.

These registers are read–only registers, which can be read at any time. Their content is not protected by any overwrite protection mechanism, so the set of four registers must be read soon after an ACCUM_INT to be sure that newer data will not cause an overwrite part way through the set. The CHx_I_PROMPT and CHx_Q_PROMPT contain the Accumulated Data from the Prompt arm. The CHx_I_TRACK and CHx_Q_TRACK contain the Accumulated Data from the Tracking arm.

To track satellites correctly, only data read with the CHx_NEW_ACCUM_DATA bit set High should be used. An overflow or underflow condition cannot be reached.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	CHx_I_TRACK[15:0] CHx_Q_TRACK[15:0] CHx_I_PROMPT[15:0] CHx_Q_PROMPT[15:0]	Bits 15:0 of the 16-bit Integrate and Dump accumulator in either I or Q parts of either Track or Prompt correlator arms	0x0000	R

Table 7.27 CORR CHx_I /_Q_TRACK/_PROMPT Register

7.6.23 CHx_SATCNTL Register - Write Address Offset <CHx_Control> + 0x00
MULTI_ SATCNTL Register - Write Address Offset 180 + 0x00
ALL_SATCNTL Register - Write Address Offset 1C0 + 0x00

The SATCNTL Register is used to set up a correlator channel to generate a particular code from the code-generator. It also is used to set-up either PRESET or UPDATE mode, and to select either SIGN0/MAG0 or SIGN1/MAG1 inputs from a RF Front-end (this feature not available on the 100-pin version of GP4020!).

CHx_SATCNTL is a write–only register that can be written into at any time. Any modification to the content is effective at the next DUMP in UPDATE mode or at the next TIC in PRESET mode for all bits, apart from PRESET UPDATEB, which defines whether a channel is in PRESET or UPDATE mode. It is important to program this register first when starting the initialisation of a PRESET sequence to get the channel into PRESET mode, or the other write operations will act too soon.

When TRACK_SEL[1:0] selects the dithering code, the Tracking arm will use the EARLY code for 20 periods of the Gold code, the LATE code for the next 20 periods and then this process of alternating between Early and Late code will be repeated indefinitely. The Tracking Arm will toggle between Early and Late Codes on every increment of a 20ms Epoch Count. Its state can be determined by reading the ACCUM_STATUS_C register.

The output code is a sequence of +1's and –1's for all code types except EARLY–MINUS–LATE where the result can also be a '0'. In EARLY–MINUS–LATE mode the values are not the +2, 0, –2 that results from the calculation (+1 or –1) – (+1 or –1), but are halved to +1, 0, –1. This must be considered when choosing thresholds in the software, as the correlation results will be exactly half of the values otherwise expected.

G2_LOAD[9:0] C/A CODE SELECTION: The CHx_SATCNTL register programs the CODE GENERATOR by setting the G2 register to the appropriate starting pattern to generate the required GPS or INMARSAT–GIC codes. The G2_LOAD register may be programmed at any time but the value is only used when the code sequence restarts, at the following DUMP in UPDATE mode, or at the following TIC in PRESET mode. The pattern to load is

the register state for the time of the second code chip. *Table 7.28 on page 79* shows the values required to select one of the 37 GPS, 19 WAAS or the 8 INMARSAT–GIC possible PRN (Pseudo Random Noise) patterns.

In UPDATE mode, the C/A code generated by the CODE GENERATOR will be changed at the DUMP following the write to CHx_SATCNTL and at this DUMP the Accumulated Data will be valid for the previous code selection. Later Dumps will be valid for the new code.

If all zeros are loaded into the G2 register, it will not clock out, and the G1 generator code will be seen on the output. This is an illegal state, which is only of use for chip testing.

Notes:

- PRN sequences 33 to 37 are reserved for non-satellite uses (e.g. Ground transmitters - "Pseudolites")
- C/A codes 34 and 37 are equivalent.
- PRN sequences 120 to 138 are selected for the Wide Area Augmentation System (WAAS).
- PRN sequences 201 to 211 are selected for INMARSAT GIC (GPS Integrity Channel) use.

Due to the initialisation of the Early–Prompt–Late shift register, all codes will always start with a ”1” for the first bit of the sequence after a Code change or a Code Slew. Subsequent cycles of the PRN sequence will be correct for the chosen satellite.

GPS PRN Signal No	G2_LOAD [9:0]	GPS PRN Signal No	G2_LOAD [9:0]	GPS PRN Signal No	G2_LOAD [9:0]
1	0x3F6	24	0x338	127	0x1E7
2	0x3EC	25	0x270	128	0x2B5
3	0x3D8	26	0x0E0	129	0x22A
4	0x3B0	27	0x1C0	130	0x10E
5	0x04B	28	0x380	131	0x12D
6	0x096	29	0x22B	132	0x215
7	0x2CB	30	0x056	133	0x337
8	0x196	31	0x0AC	134	0x0C7
9	0x32C	32	0x158	135	0x0E2
10	0x3BA			136	0x20F
11	0x374	33	0x2B0	137	0x3C0
12	0x1D0	34	0x058	138	0x029
13	0x3A0	35	0x18B		
14	0x340	36	0x316	201 GIC	0x2C4
15	0x280	37	0x058	202 GIC	0x10A
16	0x100			205 GIC	0x3E3
17	0x113	120	0x2C4	206 GIC	0x0F8
18	0x226	121	0x30A	207 GIC	0x25F
19	0x04C	122	0x1DA	208 GIC	0x1E7
20	0x098	123	0x0B2	209 GIC	0x2B5
21	0x130	124	0x3E3	211 GIC	0x10E
22	0x260	125	0x0F8		
23	0x267	126	0x25F		

Table 7.28 G2 LOAD settings required for C/A code generator, for valid PRN Numbers

Bit No.	Mnemonic	Description	Reset Value	R/W
15	GPS_NGLON	Select mode of C/A code generator. '0' = Run C/A code generator in GLONASS mode, to generate the fixed 511-bit sequence used by all GLONASS Satellites. After a reset, GPS mode is selected, but with all zeros in the G2 generator, the G1 code is seen at the output of the C/A code generator. '1' = Run C/A code Generator in GPS mode	1	W
14:13	TRACK_SEL [1:0]	Select code of Tracking arm output. '00' = Early Code '01' = Late Code '10' = Dithering code (alternate Early Code and Late Code). '11' = Early-minus-late Code	00	W
12	PRESET/ UPDATEB	Selects either PRESET or UPDATE mode. '0' = select UPDATE mode. Data updates occur at each data DUMP. '1' = PRESET mode. Data updates occur at each TIC. This bit is cleared to Low after the Preset function has been done, that is after the first TIC following the loading of the Epoch counters.	0	W
11	CODEOFF/ ONB	Test mode facility to disable the C/A Code Generator. '0' = C/A code generator Enabled. '1' = C/A code generator disabled; the Prompt, Early and Late codes are held High (code mixer outputs exactly follow inputs) and the Early–minus–late code is held LOW.	0	W
10	SOURCESEL	Selects which input source to be used by the channel, for test purposes only (*MAG1 and SIGN1 are NOT separately bonded out on 100pin GP4020 device).* '0' = selects SIGN0 and MAG0 inputs. '1' = selects SIGN1 and MAG1 inputs, via GPIO[0] (pin 100) and GPIO[1] (pin 99), when GP4020 UIM_TEST mode enabled.	0	W
9:0	G2_LOAD [9:0]	C/A CODE SELECTION FUNCTION. G2 register start pattern. See *Table 7.28* for data detail.	0x000	W

Table 7.29 CORR CHx_SATCNTL Register

7.6.24 MEAS_STATUS_A Register - Read Address Offset 0x204

This register indicates if measurement data generated by any of the 12 correlator channels, which has been generated more than once since the previous read of the register, has not been read, and has hence been missed.

If this register is always read after the Code Phase Counter, it indicates whether measurement data has been missed before the last read of the Code Phase Counter. All CHx_MISSED_MEAS_DATA bits are set Low by a hardware or software reset.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:14	*Not Used*	'0' when read	-	R
13	TIC	Set HIGH at every occurrence of TIC and is cleared by reading this ACCUM_STATUS_B register. This bit can be used as a flag to the microprocessor, to time software module swapping. It is reset by a hardware master reset (NRESET ='0') but not by an MRB in RESET_CONTROL.	0	R
12	MEAS_INT	Provided that interrupts are enabled, the MEAS_INT bit is set High at each TIC and 50 ms before each TIC (if the TIC period is greater then 50 ms), and is cleared by reading this register. This bit can be used to tell the microprocessor that new Measurement Data is available. It is reset by a hardware master reset (NRESET = '0'), but not by a software reset.	0	R

Bit No.	Mnemonic	Description	Reset Value	R/W
11	CH11_MISSED_ MEAS_DATA	'1' = one or more sets of measurement data have been missed since the last read from this register. It is set High by a read from the Code Phase Counter of the same channel, when the previous value in the Code Phase Counter has not been read, and is reset by a read from the MEAS_STATUS_A register or by disabling the channel. '0' = no missed data.	0	R
10	CH10_MISSED_ MEAS_DATA	(as bit 11 but for channel 10)	0	R
9	CH9_MISSED_ MEAS_DATA	(as bit 11 but for channel 9)	0	R
8	CH8_MISSED_ MEAS_DATA	(as bit 11 but for channel 8)	0	R
7	CH7_MISSED_ MEAS_DATA	(as bit 11 but for channel 7)	0	R
6	CH6_MISSED_ MEAS_DATA	(as bit 11 but for channel 6)	0	R
5	CH5_MISSED_ MEAS_DATA	(as bit 11 but for channel 5)	0	R
4	CH4_MISSED_ MEAS_DATA	(as bit 11 but for channel 4)	0	R
3	CH3_MISSED_ MEAS_DATA	(as bit 11 but for channel 3)	0	R
2	CH2_MISSED_ MEAS_DATA	(as bit 11 but for channel 2)	0	R
1	CH1_MISSED_ MEAS_DATA	(as bit 11 but for channel 1)	0	R
0	CH0_MISSED_ MEAS_DATA	(as bit 11 but for channel 0)	0	R

Table 7.30 CORR MEAS_STATUS_A Register

7.6.25 MULTI_CHANNEL_SELECT Register - Write Address Offset 0x1F4

This register is used to define which of the 12-correlator channels can be addressed using accesses from the "Multi Control" and "Multi Accumulate" addresses (refer to Register map). This may be used to set several channels to mostly the same conditions. This feature could be used for a parallel search for one satellite. For example:

1) operations such as setting each Carrier DCO to the same frequency;

2) adjust all selected channels by the same value, (such as a Code Slew to shift the code phases together to a new search area).

Bit No.	Mnemonic	Description	Reset Value	R/W
15:12	*Not Used*		-	W
11	CH11_SELECT	'1' = enables the Multi-channel write operations on Channel 11. '0' = disables Multi-channel write operations on Channel 11.	0	W
10	CH10_SELECT	(as bit 11 but for channel 10)	0	W
9	CH9_SELECT	(as bit 11 but for channel 9)	0	W
8	CH8_SELECT	(as bit 11 but for channel 8)	0	W
7	CH7_SELECT	(as bit 11 but for channel 7)	0	W
6	CH6_SELECT	(as bit 11 but for channel 6)	0	W
5	CH5_SELECT	(as bit 11 but for channel 5)	0	W
4	CH4_SELECT	(as bit 11 but for channel 4)	0	W
3	CH3_SELECT	(as bit 11 but for channel 3)	0	W
2	CH2_SELECT	(as bit 11 but for channel 2)	0	W
1	CH1_SELECT	(as bit 11 but for channel 1)	0	W
0	CH0_SELECT	(as bit 11 but for channel 0)	0	W

Table 7.31 CORR MULTI_CHANNEL_SELECT Register

7.6.26 PROG_ACCUM_INT Register - Write Address Offset 0x1AC

This register is used in conjunction with the INTERRUPT_PERIOD bit of the SYSTEM_SETUP register to configure the period of the Accumulation Data Interrupt (ACCUM_INT) signal. This signal is used to tell the microprocessor that it should check ALL the STATUS registers in the correlator, to see if there is any new Accumulation data. This should occur once for every DUMP event, but normally the interrupt period should be shorter than DUMP (i.e. <1.023ms).

ACCUM_INT is generated by a 13-bit binary down counter which counts down to zero, producing an ACCUM_INT output. It then loads to a Preset value stored in its Preset register and starts to count down again. If the Preset value is P, the count sequence is P, P–1, P–2, ..., 1, 0, P, P–1. Hence, the counter divides by P+1, producing an output with a period of (P+1) * clock period. Since the ACCUM_INT counter is clocked by the multi–phase clock, the clock rate is (7 * clock period) (nominally 40MHz, i.e. 25ns). The value stored in the Preset register can be modified in one of two ways:

i) Toggle the INTERRUPT_PERIOD bit of the SYSTEM_SETUP register,

ii) Writing to the PROG_ACCUM_INT location.

Either of these actions will overwrite the previous contents of the Preset value and either one or both methods may be used. If the Interrupt Counter detects an edge on the INTERRUPT_PERIOD bit it will load into the Preset register either 0x0B45 (505.05μs) if INTERRUPT_PERIOD is a '0', or 0x1313 (854μs) if INTERRUPT_PERIOD is a '1'.

Alternatively, the ACCUM_INT counter may be loaded by writing direct to the PROG_ACCUM_INT location. In this case, the new ACCUM_INT period is as follows:

ACCUM_INT Period = (PROG_ACCUM_INT + 1) * 7 /(40MHZ)

Bit No.	Mnemonic	Description	Reset Value	R/W
15:13	*Not used*		-	W
12:0	ACCUM_INT[12:0]	13-bit ACCUM_INT down-count period value.	0x0B45	W

Table 7.32 CORR PROG_ACCUM_INT Register

7.6.27 PROG_TIC_HIGH Register - Write Address Offset 0x1B4

The PROG_TIC_HIGH and PROG_TIC_LOW register locations operate in conjunction to set the period of TIC. TIC is generated by a 21-bit binary down counter when it reaches zero. It then loads to a Preset value stored in its Preset register and starts to count down again. If the Preset value is P, the count sequence is P, P–1, P–2, ...,1, 0, P, P–1. Hence, the counter divides by P+1 producing an output with a period of (P+1) * clock period. Since the TIC counter is clocked by the multi–phase clock, the clock period is (7 * clock period) (nominally 40MHz i.e. 25ns). Writing to the PROG_TIC_HIGH/_LOW locations can modify the value stored in the Preset register. This will overwrite the previous contents of the Preset value. The Preset value is set to be 0x08B823, which gives a nominal TIC period of 0.0999999seconds exactly (i.e. 100ns short of 100.0000ms, derived by {(571427+1) * 7 / 40MHz}).

The TIC counter may be loaded by writing directly to the PROG_TIC locations. This may be achieved in one of two ways:

i) PROG_TIC_HIGH value can be written, followed by the PROG_TIC_LOW value, (at which point the full 21-bits are transferred to the Preset register);

ii) PROG_TIC_LOW value may be written to modify the lower 16-bits of the Preset value.

It should be noted that in the former case, the top 5-bits programmed as PROG_TIC_HIGH are stored locally to the TIC counter. Even if a write to PROG_TIC_LOW does not directly follow the write to PROG_TIC_HIGH, the next PROG_TIC_LOW write will still transfer all 21-bits. It is also necessary to ensure that the write to PROG_TIC_HIGH precedes the write to PROG_TIC_LOW, rather than follows it.

The transfer of data to the TIC counter data latches occurs under control of the multi–phase clock write cycle and the write to the Preset register happens subsequent to the main internal write.

Using the PROG_TIC write locations the TIC period is as follows:

TIC Period = ((PROG_TIC_HIGH * 65536) + PROG_TIC_LOW + 1) * 7 / (40MHZ)

Bit No.	Mnemonic	Description	Reset Value	R/W
15:5	*Not used*		-	W
4:0	PROG_TIC[20:16]	Bits 20:16 of the 21-bit TIC Period Register. Must be written before PROG_TIC_LOW values.	0x08	W

Table 7.33 CORR PROG_TIC_HIGH Register

7.6.28 PROG_TIC_LOW Register - Write Address Offset 0x1BC

This register contains the 16 least significant bits for the PROG_TIC register. Refer to "PROG_TIC_HIGH" for more information.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	PROG_TIC[15:0]	Bits 15:0 of the 21-bit TIC Period Register	0xB823	W

Table 7.34 CORR PROG_TIC_LOW Register

7.6.29 RESET_CONTROL Register - Write Address Offset 0x1FC

This register is used to disable correlator channels which are not required in the navigation solution, and at the same time undertake a full hardware reset of the channel. By removing multiphase clocks from a disabled channel, the power-consumption of the 12-channel correlator block can be reduced.

When a CHx_RSTB bit is set Low to disable a correlator channel, the reset bit inhibits propagation of the clock phases to the CHx tracking channel. It also resets the Accumulated Data flags, Code DCO and Carrier DCO accumulators, the I & Q accumulators, and the Code Phase Counter. A CHx_RSTB does not reset the Carrier

Cycle, Code Slew or the Epoch counters. At the end of the reset, the channel enable resets the code generator to a previously programmed start phase. This is all required for the parallel search algorithm of one satellite signal using many channels in order to start from a known relative code-phase on all the channels. All of the control registers in CHx can be programmed and read as usual during the reset state. To restart normal operation in several different channels at the same time, the corresponding CHx_RSTB bits should be set to High during the same write operation. All CHx_RSTB are set Low by a master reset, (both hardware and software), so a write Low to bit 0 of this register will force a Low onto bits 12 to 1 irrespective of what was on the bus.

Setting CHx_RSTB to Low when a channel is not required can reduce power consumption.

When the MRB bit is set Low (a software reset), the effect is similar to a hardware reset. However the clock generator, the time-base generators, and measurement data registers are not affected, and the Status bits ACCUM_INT, DISCIP, DISCIP_GLITCH, MEAS_INT, and TIC are not reset. MRB should be set to High to allow access to all of the various registers. MRB is set High by a hardware reset.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:13	*Not Used*		-	W
12	CH11_RSTB	'1' = enable Channel 11. '0' = disables Channel 11. Inhibits all multiphase-clock phases to Channel 11, and resets the Accumulated Data flags, Code DCO and Carrier DCO accumulators, the I & Q accumulators, and the Code Phase Counter.	1	W
11	CH10_RSTB	(as bit 12 but for channel 10)	1	W
10	CH9_RSTB	(as bit 12 but for channel 9)	1	W
9	CH8_RSTB	(as bit 12 but for channel 8)	1	W
8	CH7_RSTB	(as bit 12 but for channel 7)	1	W
7	CH6_RSTB	(as bit 12 but for channel 6)	1	W
6	CH5_RSTB	(as bit 12 but for channel 5)	1	W
5	CH4_RSTB	(as bit 12 but for channel 4)	1	W
4	CH3_RSTB	(as bit 12 but for channel 3)	1	W
3	CH2_RSTB	(as bit 12 but for channel 2)	1	W
2	CH1_RSTB	(as bit 12 but for channel 1)	1	W
1	CH0_RSTB	(as bit 12 but for channel 0)	1	W
0	MRB	'1' = no effect '0' = activate software reset of 12-channel correlator.	1	W

Table 7.35 CORR RESET_CONTROL Register

7.6.30 STATUS Register - Write Address Offset 0x200

This register allows the bits on the Accumulation Status registers ACCUM_STATUS_A, ACCUM_STATUS_B, and ACCUM_STATUS_C to be latched for reading. This could be useful if the Accumulation data is obtained by a polling routine, rather than an interrupt driven routine.

A write operation to this location, irrespective of the data on the bus, latches the state of all status bits contained in ACCUM_STATUS_A, ACCUM_STATUS_B, and ACCUM_STATUS_C registers. Performing a write to STATUS prior to reading the status registers ensures reading of stable status values. The latch takes effect within 300ns of the trailing edge of the write pulse. The active edge transition of the ACCUM_INT signal will also latch the state of the status bits. It is not necessary to write to STATUS when the status registers are to be read as a response to the ACCUM_INT signal in an interrupt handling routine. The write to STATUS is required only when the status registers are read at times that are not synchronised to the interrupts. These two mechanisms are mutually exclusive and should not be used together – if both are used, a write to STATUS soon after the occurrence of an ACCUM_INT signal can result in confused readings. To avoid conflict the INTERRUPT_ENABLE in the SYSTEM_SETUP register should be set to Low if writes to STATUS are to be used.

If the INTERRUPT_ENABLE bit in SYSTEM_SETUP register is set to Low, the interrupt will not latch the status bits in the status registers, but a STATUS write access will do so.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	*Not used*	Write–only location provided to allow latching the state of all status bits in ACCUM_STATUS_A, ACCUM_STATUS_B, and ACCUM_STATUS_C.	0	W

Table 7.36 CORR STATUS Register

7.6.31 SYSTEM_SETUP Register - Write Address Offset 0x1F8

This register is used to set-up some top-level correlator configurations.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:11	*Not used*		-	W
10	MEAS_INT_SOURCE	'1' = MEAS_INT output cleared by a read of MEAS_STATUS_A register. '0' = MEAS_INT output cleared by a read of ACCUM_STATUS_B register.	0	W
9:8	*Not used*		-	W
7	INTERRUPT_PERIOD	'1' = set default ACCUM_INT period to 854μs. '0' = set default ACCUM_INT period to 505.05μs. *See description of PROG_ACCUM_INT for more detail.*	0	W
6	*Reserved*		0	W
5	INTERRUPT_ENABLE	Enables and disables correlator-sourced ACCUM_INT and MEAS_INT interrupt signals. '1' = enable correlator interrupts. '0' = disable correlator interrupts	0	W
4:1	DISCOP_SELECT[3:0]	Select output signals for DISCOP output: '1XXX' = 100kHz Square-wave, derived from M_CLK. '0X1X' = Channel 0 DUMP signal. Indicates when a DUMP event occurs on channel 0. '010X' = Raw_Timemark output (NOT 1pps Timemark). '0001' = High ('1') output '0000' = Low ('0') output	0000	W

7: 12-Channel Correlator

Bit No.	Mnemonic	Description	Reset Value	R/W
0	CARRIER_MIX _DISABLE	'1' = Disable Carrier mixers (Note 1). '0' = Enable Carrier mixers.	0	W

Table 7.37 CORR SYSTEM_SETUP Register

Note 1: Disable carrier mixers by driving a fixed '+1' level on the carrier DCO port on all channel carrier mixers, so that input mixer data is passed unaltered to the following code mixers.

7.6.32 TEST_CONTROL Register - Write Address Offset 0x1F0

This register is used to enable various test modes within the 12-channel correlator.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:12	*Not used*		-	W
11:9	PATH_SEL[2:0]	To allow for simple factory testing of the chip, the 12-channel correlator contains four separate scan paths, one for each of the major counters in the chip. Only one of these paths may be enabled at any time and the scan path to be used is selected via the PATH_SEL[2:0] bits as follows: '000' = Not used '001' = ACCUM_INT Counter '010' = TIC Counter '011' = 100KHz Output Counter '100' = Raw_Timemark Pulse Width Counter '1X1' = Not Used '11X' = Not Used	000	W
8	EN_SCAN_PATH	'1' = Enable Scan Test '0' = Disable Scan Test When Scan Test is enabled: DISCIP1 (GPIO[4]) becomes SCAN_IN; DISCOP becomes SCAN_OUT; MULTI_FNIO becomes SCANCLK; DISCIO becomes SCANSEL; (See Note 1)	0	W
7	*Not used*		-	W
6	TEST_CACODES	Allows checking of PROMPT C/A codes from all channels, 0 to 11. Inverted PROMPT codes for channels 11:0 available on Data bus bits [11:0], and data also seen in parallel by a read to any CH6 to CH11 read address. '1' = enable C/A code test '0' = disable C/A code test	0	W
5	TEST_DATA	This bit sets the sign of the modulation of the test data generated when TEST_SOURCE is set.	0	W
4	TEST_SOURCE	'0' = Enable self-test generator '1' = Disable self-test generator (See Note 2)	0	W

Bit No.	Mnemonic	Description	Reset Value	R/W
3	TM_TEST	Enables Tracking Module Test mode. This permits writes to the registers which are normally inhibited from write operations, namely CHx_CARRIER_CYCLE_COUNTER and CHx_CODE_PHASE_COUNTER registers. '1' = Enable Tracking Module Test. '0' = Disable Tracking Module Test.	0	W
2	FE_TEST	'1' = Enable RF Front End Test mode. '0' = Disable RF Front End Test mode. *Refer to the text at bottom of this register for more information on RF Front End Test*.	0	W
1	EN_DUMMYTICS	Enable DUMMYTICS Input. '1' = Enable DUMMYTICS input. '0' = Disable DUMMYTICS input. (See Note 3)	0	W
0	EN_DUMMYDUMP	'1' = Enable DUMMYDUMP input. '0' = Disable DUMMYDUMP input. (See Note 4)	0	W

Table 7.38 CORR TEST_CONTROL Register

Notes:

1) In EN_SCAN_PATH, the "DISCOP = SCAN_OUT" function may be over–ridden by the DISCOP_ SELECT_100KHZ function of SYSTEM_ SETUP. The MULTI_FN_IO and DISCIO pins will only connect to the signals identified in this mode if UIM_TEST mode has been set-up using TEST (pin 67(100-pin package)) and TESTMODE (pin 74 (100-pin package)). Both of these pins should be configured as inputs via the IO_REV register in the PCL block. Refer to *Section 12.7.2 "PCL Input / Output Control register - IO_REV* - Memory Offset 0x00C *on page 127*, for details.

2) Enables a self-test generator formed from the CH0 Code Generator. The data replaces the SIGN0 and MAG0 inputs. It has a chip rate and phase set by the CH0_CODE_DCO and a carrier frequency set by the CH0_CARRIER_DCO. The code is set by writing the appropriate start value into the CH0_SATCNTL register, and the CH0_SLEW_COUNTER can be programmed to delay the start of the code generation by a number of half code chips. The three most significant bits of the Carrier DCO are decoded to give the SIGN with 50% of Highs and the MAG with 25% of Highs. The polarity of the data pattern is set by TEST_DATA, EXORed with the CH0 C/A code.

3) Changes the function of the DISCIP1 input to a DUMMYTIC input. This replaces the TIC from the Timebase generator so that a TIC effect will only occur when there is a Low to High transition on DISCIP1 (derived from GPIO[4] (pin 95 (100-pin package))), to latch new Measurement Data. The DISCIP1 input must be held High for at least 200ns for each DUMMYTIC.

4) Enable DUMMYDUMP signal input. Changes the function of an internal signal (MOTINTELB) to be a DUMMYDUMP signal (MOTINTELB is derived from DISCIO input (pin 55 (100-pin package)) when UIM_TEST mode is enabled (refer to PCL documentation)). A DUMMYDUMP will operate in the same way as a normal DUMP (reset all of the code generators and transfer the contents of all integrators into the Accumulated Data registers). Each Low to High transition of DISCIO will cause a DUMMYDUMP and if DISCIO is already High when EN_DUMMYDUMP is set, one will occur immediately. Selecting Dummy dump mode does not inhibit normal DUMP events. The DISCIO pin must be held High for at least 200ns for each DUMMYDUMP.

7.6.32.1 Details of RF Front End Test mode (FE_TEST)

When FE_TEST is set High this test control forces the SIGN input to channel 11 and the MAG input to channel 5 both to Low. This allows the evaluation of the RF Front end SIGN (on channel 5) and MAG (on channel 11) duty cycles. The Front end to be tested is selected by the SOURCESEL bits in CH5_SATCNTL and CH11_SATCNTL.

7: 12-Channel Correlator

To get the SIGN and MAG count correctly into the accumulators, both the carrier and code mixers must be made transparent.

The carrier mixing may be disabled by either:

(1) Setting CARRIER_MIX_DISABLE (bit 0 in SYSTEM_SETUP) to High to force a +1 on the Carrier DCO inputs to all channels;

(2) If continued position finding is required from the other channels during the test, by setting CH5_ and CH11_CARRIER_DCO_INCR to all 0's, to give a constant level (zero frequency). This level should be set to a known value by putting channels 5 and 11 briefly into the reset state (by using RESET_CONTROL register bits 6 and 12) during the time their Carrier DCO's are programmed to zero frequency. This reset forces the phase to all 0's and hence the drives to the Prompt In–phase mixer to a fixed +1 and not a randomly selected –2, –1, +1, or +2 that would result from just setting the frequency.

The C/A code mixing must be disabled by setting CODE_OFF/ONB (bits 11 in both CH5_ and CH11_SATCNTL) to High. However, as the period of the count is set by the DUMPs from the Code Generator, the DCO clock to the Code Generator must be set to the required frequency by programming the Code DCO, even though the code output is disabled. A typical value is the frequency for the nominal code-chipping rate, so that the SIGN and MAG counts are over a millisecond.

The results of monitoring the Front–end of the receiver may be used for fault diagnosis and for tuning the parameters in the software for optimum satellite tracking with the particular Front–end or SIGN/MAG duty cycle.

To find the duty cycle of the SIGN signal, channel 5 is used. The In–phase accumulator CH5_I_PROMPT will add +1 for each SIGN sample at High and will add –1 for each SIGN sample at Low. If the duty cycle is correct at 50%, the sum will always be close to zero and only differ by the imbalance of sampling at the beginning and end of the integration period.

The duty cycle may be calculated as follows:

$$\textbf{SIGN duty cycle} = R_S = \textbf{NSIGN1 / N = (N + ACC5) / 2N} \textit{ (nominally 0.50)}$$

Where: N = Total No of samples in integration period.

NSIGN1 = Total No of samples for which SIGN was High.

NSIGN0 = Total No of samples for which SIGN was Low.

ACC5 = Total value in the CH5_I_PROMPT accumulator, as read after a DUMP.

N = N SIGN1 + N SIGN0

ACC5 = N SIGN1 – N SIGN0

To find the duty cycle of the MAG signal, channel 11 is used. The In–phase accumulator CH11_I_PROMPT will add –3 for each MAG sample at High and will add –1 for each MAG sample at Low. If the duty cycle is correct (30%), the sum will be: –1.6 * (Number of samples) plus an allowance for the imbalance of sampling at the beginning and end of the integration period. The duty cycle may be calculated as follows:

$$\textbf{MAG duty cycle, } R_M = \textbf{NMAG3 / N = – (N + ACC11) / 2N} \textit{ (nominally 0.30).}$$

Where: N = Total No of samples in integration period.

N MAG3 = Total No of samples for which MAG was High

N MAG1 = Total number of samples for which MAG was Low

ACC11 = Total value in the CH11_I_PROMPT accumulator, as read after a DUMP.

N = N MAG3 + N MAG1 ,

ACC11 = –3 * N MAG3 –N MAG1

7.6.33 TIMEMARK_CONTROL Register - Write Address Offset 0x1EC

This register is used to set-up the correlator part of the 1PPS Timemark Generator (i.e. the Raw_Timemark Generator). The RAW TIMEMARK Generator operates in one of two ways:

1) **Armed mode.** In Armed mode setting the ARM_TIMEMARK bit arms the RAW TIMEMARK generator which subsequently produces a RAW TIMEMARK output pulse coincident with the next rising edge of TIC. This then resets the ARM_TIMEMARK bit ready for a new arming sequence in the future.

2) **Free-run mode**. In Free–run mode, enabled by setting the FREE_RUN_TIMEMARK bit High, the ARM_TIMEMARK bit is disabled. A RAW TIMEMARK pulse is produced coincident with the first rising edge of TIC after the FREE_RUN_TIMEMARK bit has been set, and then on an integer number of TICs determined by the FREE_RUN_RATIO bits. In free run mode the TIMEMARK period is:

TIMEMARK Period = (FREE_RUN_RATIO + 1) * TIC Period

The RAW_TIMEMARK signal is then used by the 1PPS Timemark Generator, in conjunction with software to produce a UTC aligned 1PPS output, down to a resolution of 25ns. The RAW_TIMEMARK generator can also produce a 1PPS output without the assistance of the 1PPS Timemark generator, but the resolution of the Timemark will be 175ns which is often considered too slack for high precision time-keeping.

In the GP4020, RAW_TIMEMARK can be accessed through:

1) DISCOP, if the SYSTEM_SETUP register is configured to output RAW_TIMEMARK, and DISCOP_MUX in the PCL IO_REV register is set to output DISCOP onto GPIO[5] (pin 93 (100-pin package));
2) TIMEMARK (pin 69 (100-pin package)), if TIC_CORR[2:0] in TIC_RET register (PCL Block) is set to '000'.

Refer to Section *15 "1PPS TIMEMARK GENERATOR" on page 149* for more information.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:7	*Not used*		-	W
6:2	FREE_RUN_RATIO[4:0]	5-bit Ratio value used to set repetition rate of FREE_RUN mode Raw Timemark events, in terms of numbers of TICs. Range = 1 to 15 TICs (approx. 100ms to 1.5s, with TIC at approx. 100ms.)	0x00	W
1	FREE_RUN_TIMEMARK	'1' = Enable Free Run Timemark. Output Raw Timemark pulses in multiples of TIC events defined by FREE_RUN_RATIO. '0' = Enable Armed Timemark mode. Raw Timemark event produced on the rising edge of TIC following the setting of the ARM_TIMEMARK bit.	0	W
0	ARM_TIMEMARK	'1' = ARM RAW Timemark generator which subsequently produces a RAW TIMEMARK output pulse coincident with the next rising edge of TIC. ARM_TIMEMARK cleared by Raw Timemark event. '0' = no effect.	0	W

Table 7.39 CORR TIMEMARK_CONTROL Register

7.6.34 X_DCO_INCR_HIGH Register - Write Address Offset 0x1A4

The X_DCO_INCR_HIGH register may be used to write the high bits for any Carrier or Code DCO in any channel. A write to X_DCO_INCR_HIGH must always be followed by a write to the appropriate CHx_CARRIER_DCO_INCR_LOW or CHx_CODE_DCO_INCR_LOW to define the destination and to complete the action.

Using X_DCO_INCR_HIGH rather than CHx_CARRIER_DCO_INCR_HIGH gives a quicker way of loading the whole DCO's values because the _LOW write may follow the X_DCO_INCR HIGH write immediately (without incurring a 300ns wait). Register structure is identical to the CHx_<>_DCO_INCR_HIGH registers.

This Page intentionally left blank.

8 DMA CONTROLLER (DMAC)

The GP4020 contains a DMA controller, which assists the processor to move large blocks of data around a system. Data transfer between memory blocks, or between memory and a peripheral can be extremely cycle-intensive for a processor. The ARM processor, for example, requires at least nine clock cycles to move a word of data from one address in memory to another.

The processor with a source and a destination initialises the DMAC system for the data transfer. On receipt of a DMA request, the DMA Controller acquires control of the system address and data buses and proceeds to transfer data until a stop condition is met. The DMA Controller may then be auto-initialised or manually reprogrammed as desired.

The Base Address for the DMAC is 0xE000 C000. The GP4020 DMAC has two channels. These channels can be configured to undertake two types of data transfer.

8.1 Single-Addressed (Fly-by) Data transfers

Single-addressed (Fly-by) data transfers are possible between memory and either UARTs 1 or 2.

A Single-addressed Transfer is the faster of the two types of DMA data transfer, in that one data transaction per bus cycle can be implemented. By its very nature, a single-address may be an area of memory, implying that the other end of the transfer must be a fixed peripheral, which is signalled using a hardware handshake (dreq and dack). Refer to *Section 6.2.1 in the Firefly MF1 Core Design Manual (DM5003)* for details of Single-addressed transfers.

Each DMAC channel is capable of generating an address and hardware-acknowledge signal for a particular data transaction. Data is presented to the bus by the source and written to the destination during a single bus transaction; it is not buffered by the DMA controller. An address is broadcast onto the bus simultaneously.

In the GP4020, hardware handshake signals *dack* and *dreq* emanate from each of the two DMAC channels to control fly-by data transfers with UARTs 1 and 2 as follows:

i) DMAC Channel 1 (dack1 and dreq1) for Data from UART1 RX (received data) to Memory;

ii) DMAC Channel 1 (dack1 and dreq1) for Data from Memory to UART1 TX (transmitted data);

iii) DMAC Channel 2 (dack2 and dreq2) for Data from UART2 RX (received data) to Memory;

iv) DMAC Channel 2 (dack2 and dreq2) for Data from Memory to UART2 TX (transmitted data);

The configuration of the DMAC channels does not allow simultaneous transmit and receive from one UART in fly-by mode.

8.1.1 Set up example of DMAC for a Fly-by transfer from memory to UART TX

The following example shows the sequence of events required to program and enable the GP4020 DMAC to provide a Fly-by data transfer from an area of memory to a UART Transmit output.

1) Initialise UART1 (or 2) for data transmission:

   1.1) For the UART 1 (or 2) Serial Control Register (CR):

   1.1.1) Set to “1” the Transmit Channel Control bit (bit 1) to enable the UART 1 (or 2) transmit channel.

   1.1.2) Clear to ”0” the Clock Source bit (bit 2), to ensure the UART Clock is connected internally to the UART_CLK source, from the System Clock Generator block (SCG).

   1.1.3) Clear to ”0” the Flow Control Type bit (bit 3) of UART Serial Control Register (CR), to enable Software Flow Control. There are no Hardware Control lines (RTS, CTS) for either UART1 or UART2 bonded out on the GP4020 device.

1.1.4) Clear to “0” the Receive Interrupt Enable bit (bit 4) to disable interrupts generated when the UART receive register is full.

1.1.5) Clear to “0” the Modem Interrupt Enable bit (bit 7) and the Error Interrupt Enable bit (bit 6) to disable interrupts from a remote modem or a UART error.

1.2) Set-up the appropriate UART baud rates, data lengths, stop bits, parity, etc using the Serial Mode Register (MR) and Baud Rate Register (BRR); *ref. Table 17.1 on page 170 thru to Table 17.11 on page 174* for settings.

2) Set-up the source of DMAC Triggering (i.e. the prompt that initiates the DMA transfers following the DMAC program cycle). The GP4020 DMAC can take triggers from both software and Hardware sources. For DMA Fly-by transfers using UART 1 as a peripheral, DMAC Channel 1 must be used. Similarly if using UART 2 as a peripheral, DMAC Channel 2 must be used.

2.1) If software triggering is required (and not hardware triggering), this can be programmed explicitly into the DMAC (refer to *Section 8.3 "DMAC Triggering" on page 99*).

2.2) DMAC Channel 1 has the flexibility of being able to undertake Fly-by transfers using Hardware triggering from a number of different sources. The trigger source required can be selected by setting up the DMAC Trigger Source bits within the System Configuration Register (SCR) (*Address 0xE000 2004*) in the System Services Module (SSM). In most cases, for fly-by transfers, it is most appropriate to use UART 1 as the DMAC Trigger Source. However, the options shown in *Table 8.1 Hardware Trigger Source selection for DMAC Channel 1" below* add flexibility to this.

SCR[5:2]	DMAC Channel 1 Trigger Source
0011	UART 2 Transmit / Receive. *Function is determined by UART 2 interrupt type.*
0101	SYSTIC 1A interrupt *(see note 1)*
0111	UART 1 Receive
1001	RF_PLL_LOCK interrupt
1011	EXTINT2 input interrupt
1101	SYSTIC 2A interrupt *(see note 1)*
1111	UART 1 Transmit

Table 8.1 Hardware Trigger Source selection for DMAC Channel 1

Note 1: Refer to *Section 7 of the Firefly MF1 Core Design Manual (DM5003)* for details of how to employ the Firefly TIC (SYSTIC) timer function. This can essentially provide regular time intervals, from which the DMAC can then trigger.

2.3) DMAC Channel 2 can only receive DMAC hardware triggers from UART 2, and no other source. Consequently, the only trigger option avails listed in *Table 8.1 above* do not exist for UART 2 DMAC Fly-by transfers.

3) Put DMAC into “Program Mode” to allow DMA commands to be programmed into DMAC before execution. This is done by clearing to ”0” the Channel Status bit (bit 0) of the Channel and Control Status Register (CSR) for the relevant DMAC Channel (UART1 uses Channel 1, UART 2 uses Channel 2).

This allows the DMAC to be programmed with commands, and DMAC operations are suspended until bit 0 is set to a “1”.

Program the DMAC Channel for a Write data transfer from memory to UART 1 (or 2) for TX:

3.1) For the DMAC Channel 1 (or 2) Control and Status Register (CSR):

3.1.1) Set to ”1” the DMAC Hardware Request Status bit (bit 1), to allow Hardware requests (*dreq* and *dack*) from the UART to control the DMAC function.

3.1.2) Clear to “0” the DMAC Software Request bit (bit 2), to disable the Software DMA transfer triggering. Note that when a software trigger is required after the DMAC is programmed, a write of a "1" to this register bit will initiate a DMA transfer.

3.1.3) Clear to “0” the DMAC Hardware Request Polarity bit (bit 3) and Hardware Acknowledge Polarity (bit 4), to indicate that the Fly-by Dreq and Dack signals from the DMAC to UARTs 1 and 2 are active High signals.

3.1.4) Set to “1” the DMAC Hardware Acknowledge Status bit (bit 5), to enable the Dack signals from UARTs 1 and 2 to indicate when either UART1 or UART2 have been implicitly addressed by the DMAC.

3.1.5) Clear to “0” the Address Mode bit (bit 6), to allow the single-addressed location in memory to be dynamically modified between successive DMA transfers. The Address Direction bit (bit 7) should be set depending on whether the address location should dynamically increment (bit 7 set to “1”) or decrement (bit 7 cleared to “0”).

3.1.6) Clear to “0” the Transfer Direction bit (bit 8), to signify data is being read from memory.

3.1.7) Clear to “0” the Transfer Mode bit (bit 9), to signify that Data transfers will be Single-addressed (i.e. Fly-by) between memory and a hardware hand-shaked peripheral (i.e. UART 1 if CSR is for DMAC Channel 1 and UART 2 if CSR is for DMAC Channel 2)

3.1.8) Clear to ”0” the Transfer Type bit (bit 10) to allow Packet Data Transfers to occur.

Packet Data transfers must be used in Single-addressed transfers with the GP4020 UARTs, as the UARTs do not have the bandwidth to cope with data transfers at full BμILD bus bandwidth (28MHz x 4 = 112Mbytes/second). In addition, if running the ARM7TDMI simultaneously with a DMA transfer, packet transfers of one word per packet will allow the ARM to operate on alternate bus cycles. DMA Block transfers will stall the ARM7TDMI, which could be fatal in a GPS system where correlator interrupts MUST be serviced.

3.1.9) Clear to “0” the Request Trigger Type bit (bit 11), to allow enable “Edge-Triggered” Packet Transfers. *Refer to Section 6.2.1.4 in the Firefly MF1 Core Design Manual* for details of Edge-triggered Packet Transfers.

3.1.10) Clear the DMAC Operand Size (bits [13:12]) to zero (i.e. 0y00), to set Byte-wide data in the DMA transfer. The GP4020 UARTs will cope with byte-wide data only.

3.1.11) It may be desirable to set-up an Interrupt Service Routine to run in the ARM7TDMI to service a DMAC Interrupt signal into the Firefly INTC Channel 3 (Refer to Section *10 "INTERRUPT CONTROLLER (INTC)" on page 107* for information on Interrupt settings). The DMAC interrupt can be generated under the conditions indicated in *Section 6.2.3.5 of the Firefly MF1 Core Design Manual*. If the Interrupt into the Interrupt Controller (INTC) in Firefly is required, Set to "1" the Interrupt Enable bit (bit 16) of the DMAC CSR; if NOT required, Clear this bit to "0".

3.1.12) Clear to “0” the Bus Lock bit (bit 17), to prevent the DMAC from being interrupted during a transfer from a Higher priority BμILD Bus Master (e.g. ARM7TDMI, SSM)

3.1.13) Clear to “0” the Peripheral Location bit (bit 18), to indicate that the UART peripheral is an Internal device to the GP4020.

3.2) Set DMAC Packet Size (bits [7:0]) of the Packet Size Register (PSR) to zero (i.e. 0x00). This signifies that each DMAC data packet will be one word in size.

3.3) Set the DMAC Base Address Register (BAR) with the base memory-location of the data needing to be transferred to the UART. With the Address Mode set to “dynamic”, this base-address should be an area of the memory map where there is a contiguous memory space (i.e. SRAM or ROM).

3.4) Set the DMAC Base Transfer Count Register (BTR), to indicate to the DMAC how many transfers are required in the DMA operation being programmed. In the case of the Packet transfer being defined here, this number is the (number of data bytes - 1), of Packet size "1" which are required to be transferred from memory to the UART 1 or 2 transmit port. So if the transfer is to be for 10,000 8-bit bytes, the setting for this register should be "10,000 - 1" = "9,999" = 0x270F.

4) Once all the DMAC features have been programmed:

4.1) Set to ”1” the Transmit Interrupt Enable bit (bit 5) of the UART 1 (or 2) Serial Control Register (CR) to enable interrupts generated when the UART Transmit register is empty. This is vital when using UART2 with hardware DMAC triggers from UART2, to ensure that the DMAC is triggered correctly (refer to *Section 8.3 "DMAC Triggering" on page 99*).

4.2) Set to "1" the DMAC Channel Status bit (bit 0) in the Channel 1 (or 2) Control and Status Register (CSR), to allow the DMA transfer to be triggered as defined in step 2) above. This action should be done independently of any other settings to the Control and Status Register in order to avoid setting and triggering errors.

Note: A write of a bit to the DMAC CSR register involves writing a byte, half-word or word. It is worth ensuring that the settings already programmed into the CSR are not corrupted when setting bit 0 to "1", by re-writing the values of all the other bits in the register to those defined above.

Refer to *Section 8.3 "DMAC Triggering" on page 99* for information of how both Software and Hardware triggering operates with a DMA transfer.

8.1.2 Set up example of DMAC for a Fly-by transfer from UART RX to memory

The following example shows the sequence of events required to program and enable the GP4020 DMAC to provide a Fly-by data transfer from a UART Receiver input to an area of memory.

1) Initialise UART1 (or 2) for data reception:

   1.1) For the UART 1 (or 2) Serial Control Register (CR):

   1.1.1) Set to ”1” the Receive Channel Control bit (bit 0) to enable the UART 1 (or 2) receive channel.

   1.1.2) Clear to ”0” the Clock Source bit (bit 2), to ensure the UART Clock is connected internally to the UART_CLK source, from the System Clock Generator block (SCG).

   1.1.3) Clear to ”0” the Flow Control Type bit (bit 3) of UART Serial Control Register (CR), to enable Software Flow Control. There are no Hardware Control lines (RTS, CTS) for either UART1 or UART2 bonded out on the GP4020 device.

   1.1.4) Clear to “0” the Transmit Interrupt Enable bit (bit 5) to disable interrupts generated when the UART transmit register is empty.

   1.1.5) Clear to “0” the Modem Interrupt Enable bit (bit 7) and the Error Interrupt Enable bit (bit 6) to disable interrupts from a remote modem or a UART error.

   1.3) Set-up the appropriate UART baud rates, data lengths, stop bits, parity, etc using the Serial Mode Register (MR) and Baud Rate Register (BRR); *ref. Table 17.1 on page 170 thru to Table 17.11 on page 174* for settings.

   1.4) Set-up the source of DMAC Triggering (i.e. the prompt that initiates the DMA transfers following the DMAC program cycle). The GP4020 DMAC can take triggers from both software and Hardware sources. For DMAC Fly-by transfers using UART 1 as a peripheral, DMAC Channel 1 must be used. Similarly if using UART 2 as a peripheral, DMAC Channel 2 must be used.

   2.1) If software triggering is required (and not hardware triggering), this can be programmed explicitly into the DMAC (refer to *Section 8.3 "DMAC Triggering" on page 99*).

   2.2) DMAC Channel 1 has the flexibility of being able to undertake Fly-by transfers using Hardware triggering from a number of different sources. The trigger source required can be selected by setting up the DMAC Trigger Source bits within the System Configuration Register (SCR) (*Address 0xE000 2004*) in the System Services Module (SSM). In most cases, for fly-by transfers, it is most appropriate to use UART 1 as the DMAC Trigger Source. However, the options shown in *Table 8.1 Hardware Trigger Source selection for DMAC Channel 1" above* add flexibility to this.

   2.3) DMAC Channel 2 can only receive DMAC hardware triggers from UART 2, and no other source. Consequently, the trigger options listed in *Table 8.1 above* do not exist for UART 2 DMAC Fly-by transfers.

2) Put DMAC into “Program Mode” to allow DMAC commands to be programmed into DMAC before execution. This is done by clearing to ”0” the Channel Status bit (bit 0) of the Channel and Control Status Register (CSR) for the relevant DMAC Channel (UART1 uses Channel 1, UART 2 uses Channel 2).

   This allows the DMAC to be programmed with commands, and DMAC operations are suspended until bit 0 is set to “1”.

   Program the DMAC Channel for a Write data transfer from memory to UART 1 (or 2) for TX:

   3.1) For the Channel 1 (or 2) Control and Status Register (CSR):

   3.1.1) Set to ”1” the DMAC Hardware Request Status bit (bit 1), to allow Hardware requests (*dreq* and *dack*) from the UART to control the DMAC function.

3.1.2) Clear to “0” the DMAC Software Request bit (bit 2), to disable the Software DMA transfer triggering. Note that when a software trigger is required after the DMAC is programmed, a write of a "1" to this register bit will initiate a DMA transfer.

3.1.3) Clear to “0” the DMAC Hardware Request Polarity bit (bit 3) and Hardware Acknowledge Polarity (bit 4), to indicate that the Fly-by Dreq and Dack signals from DMAC to UARTs 1 and 2 are active High signals.

3.1.4) Set to “1” the DMAC Hardware Acknowledge Status bit (bit 5), to enable the Dack signals from UARTs 1 and 2 to indicate when either UART1 or UART2 have been implicitly addressed by the DMAC.

3.1.5) Clear to “0” the Address Mode bit (bit 6), to allow the single-addressed location in memory to be dynamically modified between successive DMA transfers. The Address Direction bit (bit 7) should be set depending on whether the address location should dynamically increment (bit 7 set to “1”) or decrement (bit 7 cleared to “0”).

3.1.6) Set to “1” the Transfer Direction bit (bit 8), to signify data is being written to memory.

3.1.7) Clear to “0” the Transfer Mode bit (bit 9), to signify that Data transfers will be Single-addressed (i.e. Fly-by) between memory and a hardware hand-shaked peripheral (i.e. UART 1 if CSR is for DMAC Channel 1 and UART 2 if CSR is for DMAC Channel 2)

3.1.8) Clear to ”0” the Transfer Type bit (bit 10) to allow Packet Data Transfers to occur.

Packet Data transfers must be used in Single-addressed transfers with the GP4020 UARTs, as the UARTs do not have the bandwidth to cope with data transfers at full BμILD bus bandwidth (28MHz x 4 = 112Mbytes/second). In addition, if running the ARM7TDMI simultaneously with a DMA transfer, packet transfers of one word per packet will allow the ARM to operate on alternate bus cycles. DMAC Block transfers will stall the ARM7TDMI, which could be fatal in a GPS system where correlator interrupts MUST be serviced.

3.1.9) Clear to “0” the Request Trigger Type bit (bit 11), to allow enable “Edge-Triggered” Packet Transfers. *Refer to Section 6.2.1.4 in the Firefly MF1 Core Design Manual for details of Edge-triggered Packet Transfers*.

3.1.10) Clear to "00" the DMAC Operand Size (bits [13:12]), to set Byte-wide data in the DMA transfer. The GP4020 UARTs will cope with byte-wide data only.

3.1.11) It may be desirable to set-up an Interrupt Service Routine to run in the ARM7TDMI to service a DMAC Interrupt signal into the Firefly INTC Channel 3 (Refer to Section *10 "INTERRUPT CONTROLLER (INTC)" on page 107* for information on Interrupt settings). The DMAC interrupt can be generated under the conditions indicated in *section 6.2.3.5 of the Firefly MF1 Core Design Manual (DM5003)*. If the Interrupt into the Interrupt Controller (INTC) in Firefly is required, Set to "1" the Interrupt Enable bit (bit 16) of the DMAC CSR; if NOT required, Clear this bit to "0".

3.1.12) Clear to “0” the Bus Lock bit (bit 17), to prevent the DMAC from being interrupted during a transfer from a Higher priority BμILD Bus Master (e.g. ARM7TDMI, SSM)

3.1.13) Clear to “0” the Peripheral Location bit (bit 18), to indicate that the UART peripheral is an Internal device to the GP4020.

3.2) Set DMAC Packet Size (bits [7:0]) of the Packet Size Register (PSR) to zero (i.e. 0x00). This signifies that each DMAC data packet will be one word in size.

3.3) Set the DMAC Base Address Register (BAR) with the base memory-location of the where the data acquired from the UART will be written to. With the Address Mode set to “dynamic”, this base-address should be an area of the memory map where there is a contiguous memory space (i.e. SRAM).

3.4) Set the DMAC Base Transfer Count Register (BTR), to indicate to the DMAC how many transfers are required in the DMA operation being programmed. In the case of the Packet transfer being defined here,

this number is the (number of data bytes - 1), of Packet size "1" which are required to be transferred from memory to the UART 1 or 2 transmit port. So if the transfer is to be for 10,000 8-bit bytes, the setting for this register should be "10,000 - 1" = "9,999" = 0x270F.

3) Once all the DMAC features have been programmed:-

4.1) Set to ”1” the Receive Interrupt Enable bit (bit 4) of the UART 1 (or 2) Serial Control Register (CR) to enable interrupts generated when the UART Transmit register is empty. This is vital when using UART2 with hardware DMAC triggers from UART2, to ensure that the DMAC is triggered correctly (refer to *Section 8.3 "DMAC Triggering" on page 99*).

4.2) Set to “1” the DMAC Channel Status bit (bit 0) in the Channel 1 (or 2) Control and Status Register (CSR), to allow the DMA transfer to be triggered as defined in step 2) above. This action should be done independently of any other settings to the Control and Status Register in order to avoid setting and triggering errors.

Note: A write of a bit to the DMAC CSR register involves writing a byte, half-word or word. It is worth ensuring that the settings already programmed into the CSR are not corrupted when setting bit 0 to "1", by re-writing the values of all the other bits in the register to those defined above.

Refer to *Section 8.3 "DMAC Triggering" on page 99* for information of how both Software and Hardware triggering operates with a DMA transfer.

8.2 Dual-Addressed (Buffered) Data Transfers

Dual Addressed (Buffered) data transfers are possible between any two memory locations within the GP4020.

The DMAC can use both of its channels to undertake a Dual-addressed transfer from one memory location to another memory location. A Dual-addressed transfer does not use the Hardware controls used with the UART peripherals, and so any two memory-mapped components within the GP4020 can use the DMAC Dual Addressed Data Transfer.

The higher priority channel (DMAC Channel 1) is used as the “Read” channel, and the lower priority channel of the pair (DMAC Channel 2) becomes the Write channel.

Each Dual-Addressed Data Transfer requires two bus transactions: the first to read the data from the source to a 1-word deep buffer, and the second to write the data to the destination. Refer to *Section 6.2.2 in the Firefly MF1 Core Design Manual (DM5003)* for details of Dual-addressed transfers.

8.2.1 Set up example of DMAC for a Dual-Addressed transfer between two memory locations.

The following text shows an example of the sequence of events required to program and enable the GP4020 DMAC to provide a Dual-Addressed Data transfer between two contiguous areas of memory. A Dual-addressed transfer uses both DMAC channels simultaneously, one to read from one location into a "buffer", and the other to write to another location.

1) Set-up the source of DMAC Triggering (i.e. the prompt that initiates the DMA transfers following the DMAC program cycle). The GP4020 DMAC can take triggers from both software and Hardware sources.

1.1) If software triggering is required (and not hardware triggering), this can be programmed explicitly into the DMAC (refer to *Section 8.3 "DMAC Triggering" on page 99*).

1.2) If hardware triggering is required, the trigger source can be selected by setting up the DMAC Trigger Source bits within the System Configuration Register (SCR) (*Address 0xE000 2004*) in the System Services Module (SSM). Refer to *Table 8.1 Hardware Trigger Source selection for DMAC Channel 192*. Note that Hardware triggering is not normally appropriate for Dual Addressed Transfers.

1.3) DMAC Channel 2 can only receive DMAC hardware triggers from UART 2, and no other source. Consequently, the trigger options listed in *Table 8.1* do not exist for UART 2 DMAC Fly-by or dual-addressed transfers.

2) Put DMAC into “Program Mode” to allow DMAC commands to be programmed into DMAC before execution. This is done by clearing to ”0” the Channel Status bit (bit 0) of the Channel and Control Status Register (CSR) for the relevant DMAC Channel (UART1 uses Channel 1, UART 2 uses Channel 2).

This allows the DMAC to be programmed with commands, and DMAC operations are suspended until bit 0 is set to “1”.

Both DMAC channels need to be programmed up at this stage. In this example, we shall use the idea that DMAC Channel 1 is used to read from memory, and Channel 2 is used to write to memory:

2.1) For both the Channel 1 and Channel 2 Control and Status Register (CSR):

2.1.1) Clear to ”0” the DMAC Hardware Request Status bit (bit 1), to disable Hardware requests (*dreq* and *dack*) from either UART 1 or UART 2 to control the DMAC function.

2.1.2) Clear to “0” the DMAC Software Request bit (bit 2), to disable the Software DMA transfer triggering. Note that when a software trigger is required after the DMAC is programmed, a write of a "1" to this register bit will initiate a DMA transfer.

2.1.3) Clear to “0” the DMAC Hardware Acknowledge Status bit (bit 5), to disable the Dack signals from UARTs 1 and 2.

2.1.4) Clear to “0” the Address Mode bit (bit 6), to allow the dual-addressed location in memory to be dynamically modified between successive DMA transfers. The Address Direction bit (bit 7) should be set depending on whether the address location should dynamically increment (bit 7 set to “1”) or decrement (bit 7 cleared to “0”).

2.1.5) In the DMAC Channel 1 Control and Status Register (CSR), clear to “0” the Transfer Direction bit (bit 8), to signify data is being read from memory. In DMAC Channel 2 Control and Status Register (CSR), set to “1” the Transfer Direction bit (bit 8), to signify data is being written to memory.

2.1.6) Set to “1” the Transfer Mode bit (bit 9), to signify that Data transfers will be Dual-addressed.

2.1.7) Clear to ”0” the Transfer Type bit (bit 10), to allow Packet Data Transfers to occur.

Packet Data transfers must be used in Dual-addressed transfers if running the ARM7TDMI simultaneously with a DMA transfer. Packet transfers of one word per packet will allow the ARM to operate on alternate bus cycles. DMAC Block transfers will stall the ARM7TDMI, which could be fatal in a GPS system where correlator interrupts MUST be serviced.

2.1.8) Clear to “0” the Request Trigger Type bit (bit 11), to allow enable “Edge-Triggered” Packet Transfers. *Refer to Section 6.2.1.4 in the Firefly MF1 Core Design Manual for details of Edge-triggered Packet Transfers*.

2.1.9) Set the DMAC Operand Size (bits [13:12]), to set the required word-width. *Refer to Table 6-3 in the Firefly MF1 Core Design Manual for details of Operand size settings.*

2.1.10) It may be desirable to set-up an Interrupt Service Routine to run in the ARM7TDMI to service a DMAC Interrupt signal into the Firefly INTC Channel 3 (Refer to Section *10 "INTERRUPT CONTROLLER (INTC)" on page 107* for information on Interrupt settings). The DMAC interrupt can be generated under the conditions indicated in *section 6.2.3.5 of the Firefly MF1 Core Design Manual (DM5003)*. If the Interrupt into the Interrupt Controller (INTC) in Firefly is required, Set to "1" the Interrupt Enable bit (bit 16) of the DMAC CSR; if NOT required, Clear this bit to "0".

2.1.11) Clear to “0” the Bus Lock bit (bit 17), to prevent the DMAC from being interrupted during a transfer from a Higher priority BμILD Bus Master (e.g. ARM7TDMI, SSM)

2.1.12) Clear to "0" the Peripheral Location bit (bit 18), to indicate that the data buffer for a dual-addressed transfer is internal to the DMAC.

2.2) Set DMAC Packet Size (bits [7:0]) of the Packet Size Register (PSR) to zero (i.e. 0x00). This signifies that each DMAC data packet will be one word in size.

2.3) Set the DMAC Base Address Register (BAR) for DMAC Channel 1 with the base memory-location of the where the data to be copied exists (e.g. EPROM). Also, set the same register for DMAC Channel 2 with the base memory location of where the copied data needs to be written to (e.g. SRAM or FLASH). With the Address Mode set to “dynamic”, this base-address should be an area of the memory map where there is a contiguous memory space.

2.4) Set the DMAC Base Transfer Count Register (BTR) of the DMAC Read channel (channel 1), to indicate to the DMAC how many transfers are required in the data transfer operation being programmed. In the case of the Packet transfer being defined here, this number is the (number of data bytes - 1), of Packet size "1" which are required to be transferred between the two memory locations previously defined. So if the transfer is to be for 10,000 8-bit bytes, the setting for this register should be "10,000 - 1" = "9,999" = 0x270F.

3) Once all the DMAC features have been programmed:-

3.1) Set to “1” the DMAC Channel Status bit (bit 0) in both the Channel 1 and 2 Control and Status Register (CSR), to allow the DMA transfer to be triggered as defined in step 2) above. This action should be done independently of any other settings to the Control and Status Register in order to avoid setting and triggering errors.

Note: A write of a bit to the DMAC CSR register involves writing a byte, half-word or word. It is worth ensuring that the settings already programmed into the CSR are not corrupted when setting bit 0 to "1", by re-writing the values of all the other bits in the register to those defined above.

Refer to Section *8.3 below* on DMAC Triggering to get an indication of how both Software and Hardware triggering operates with a DMA transfer.

8.3 DMAC Triggering

8.3.1 Hardware Triggering

Hardware Triggering of a DMAC channel is the normal mode used in Single-addressed (Fly-by) data transfers.

The configuration of the DMAC Hardware Triggering for DMAC Channel 1 occurs by setting bits [5:3] of the System Configuration Register (SCR) in the System Services Module (SSM) to any of the values in *Table 8.1 on page 92*. A packet of data, as defined within the DMAC Packet Size Register (PSR), will be transferred every time a Hardware Trigger is received. This could be when any of the interrupt events in *Table 8.1 on page 92* occurs.

For UART Transmit Triggers, this occurs when the UART Transmit Buffer is empty and waiting for more data.

For UART Receive Triggers, this occurs when the UART Receive Buffer is full and waiting for data to be read.

Transfers will continue on each Hardware Trigger event until the amount of data packets defined by the DMAC Base Transfer Count Register (BTR) have been transferred. The length of hardware trigger will determine how many triggers are needed to transfer the BTR Count. In the case of the UART, a new Hardware trigger will be produced for every packet of data transferred.

DMAC Channel 2 can only be hardware triggered from UART2, and no other source. The DMAC trigger is actually configured by the type of interrupt set-up in the UART 2 either on Transmit Buffer Empty, or Receive Buffer Full.

8.3.2 Software Triggering

Software Triggering of a DMAC channel is the normal mode used in Dual-Addressed (Buffered) data transfers.

A Software Trigger is enabled and disabled dependent on the level of bit [2] of the DMAC Channel Control and Status Register (CSR). This bit can be set to "1" to initiate a DMAC Software Trigger irrespective of whether the DMA is in Program state (CSR bit zero cleared to "0") or Ready state (CSR bit zero set to "1").

- If the DMAC Channel Status (bit 0) is in Program state (i.e. cleared to "0"), and the Software Request (bit 2) is set to "1", DMA transfers will begin immediately after the Channel Status is set to Ready (bit "0" set to "1").

- If the DMAC Channel Status (bit 0) is in Ready state (i.e. set to "1"), and the Software Request (bit 2) is set to "1", DMA transfers will begin immediately.

In both cases, one software trigger will allow the DMAC channel to transfer the number of packets defined in the Base Transfer Count Register (BTR) through to completion. The Software Request can be terminated by writing "0" to CSR bit 2.

However, this can only occur if the DMAC channel is configured to undertake a "Packet" transfer, as a block transfer will give no allocation of the BμILD bus to the ARM7TDMI. Otherwise, when the number of transfers defined in the Base Transfer Count Register (BTR) has been completed, the DMAC will clear the Software Request bit (CSR bit 2) to "0".

8.4 Cautionary Notes

8.4.1 Packet Transfers in place of Block Transfers

For Both Single-addressed and Dual-addressed transfers using the GP4020 DMA, it is NOT recommended to use DMA Block-transfers, but to use Packet transfers instead.

Packet transfers allow the ARM to have access to the BμILD bus potentially on alternate clock cycles. When a Block transfer is initiated, the DMAC occupies the BμILD bus continuously for the period it takes to transfer the whole block of data defined by the contents of the Base Transfer Count Register in the DMA. In a GPS receiver, the RTOS running on the ARM microprocessor needs to access the 12-channel correlator to service ACCUM_INT interrupts, once every 500us or so. A DMA block transfer may prevent the ARM from accessing the correlator in this time-critical activity, and the GPS function may fail consequently.

8.4.2 ARM FIQ Promotion

The ARM7TDMI has the lowest priority on the GP4020 BμILD bus, under the DMA and SSM blocks (refer to *Section 2.5 in the Firefly MF1 Core Design Manual (DM5003)* for more information). When the DMAC is undertaking a data transfer, the DMAC has priority over the ARM7TDMI.

It is NOT recommended to prioritise the ARM7TDMI over the DMAC during a DMA transfer when the ARM receives a FIQ interrupt, although a mechanism exists for this. This is achieved by setting to "1" the "Bus Arbitration Bit" (bit 1) of the System Configuration Register in the Firefly SSM, to enable ARM FIQ promotion (*refer to Table 2-5 in the Firefly MF1 Core Design Manual (DM5003)*). Problems arise in the DMA, when the DMAC is interrupted from its transfer, and it can cause the DMAC functions to crash when the BμILD bus is handed back to it.

Further details for the programming of the DMAC Controller can be found in *Section 6 of the "Firefly MF1 Core Design Manual" DM5003*, available from Zarlink Semiconductor.

This Page intentionally left blank.

9 GENERAL PURPOSE INPUT OUTPUT (GPIO) INTERFACE

9.1 Introduction

A set of 8 general purpose static input output logic lines are included in the GP4020 to allow multiple static data to be provided to external features, or allow multiple input data lines to be read. The GPIO pins are multiplexed by the Peripheral Control Logic (PCL) between the BμILD Serial Input Output (BSIO) lines, and control lines to the Correlator. Refer to *Section 6 "BμILD SERIAL INPUT OUTPUT (BSIO) INTERFACE" on page 33* and *Section 12.4 "Multiplex Logic" on page 119*, for further information.

The GPIO interfaces internally on the GP4020 to the BμILD bus, and provides eight Input / Output pins to the outside world. A block diagram of the GPIO block appears in *Figure 9.1 below*.

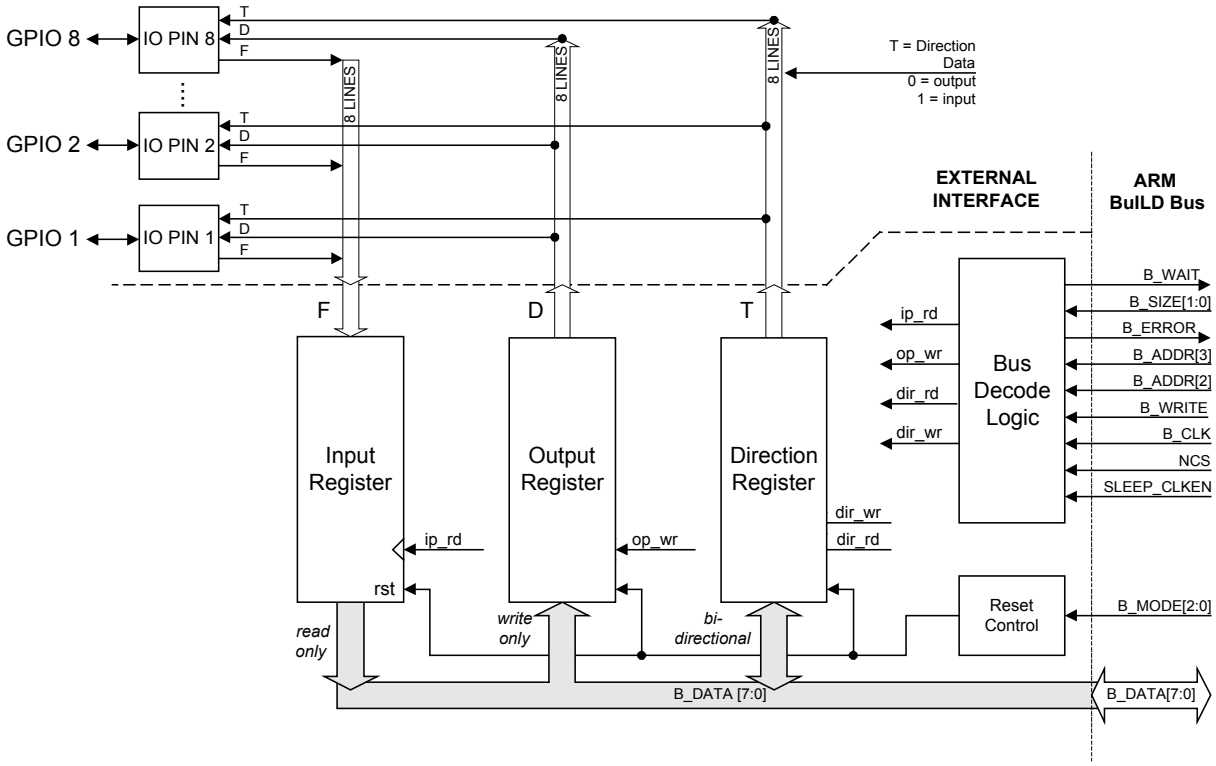


Figure 9.1 GPIO Block Diagram

Each of the eight IO pins has three data / control lines associated with it:

F - Input data

D - Output data

T - Tristate output enable/disable. Used to set the data direction into / out of the pin.

A block-diagram of the bi-directional Input / Output interface pin electronics appears in *Figure 9.2 below*.

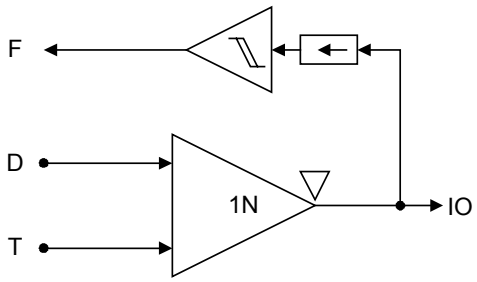


Figure 9.2 GPIO Pad Cell Configuration

The GPIO module must be read or written in 32-bit accesses, although only the lower eight bits of the BμILD data bus (b_data[7:0]) are used. Data is written to registers on the falling edge of B_CLK. Data is read from registers on a high (logic '1') value of B_CLK (see diagram BμILD Bus Interface). During the read operation the remaining 24-bits of the 32-bit bus are not driven, and will be held to a non-floating value by the bus hold cells (a chip-wide resource external to this module).

The nCS signal goes low if a memory access occurs in the address range 0xE000 5000 to 0xE000 5FFF, that is the I/O area allocated to the GIO. The bus decoding only tests B_ADDR[3] and B_ADDR[2], so multiple reflection of the four valid (three actually used) 32-bit address spaces will occur over this range. *Figure 9.3 below* shows the timing of the BμILD Bus interface to the GPIO for both Data Read and Data Write cycles.

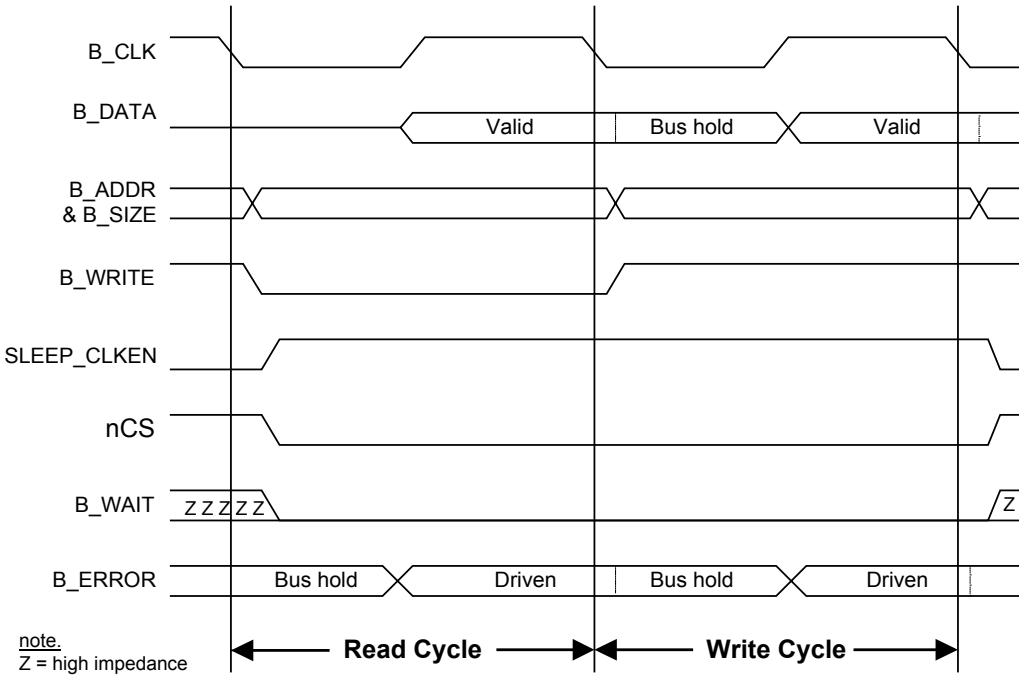


Figure 9.3 GPIO BμILD Bus interface timing

The B_ERROR signal goes high if an illegal access occurs. An illegal access means a read from a write only register or unused location, or a write to a read only register or unused location. In addition, if 8-bit or 16-bit accesses occur B_ERROR will be asserted, this is to assist software error checking. Whilst B_ERROR is being asserted data writes are disabled and will not modify registers, but read operations will still drive the bus with data if a readable register exists at that location.

B_SIZE[1:0]	Data size	B_ERROR
00	8-bit	Error, bus error asserted
01	16-bit	Error, bus error asserted
10	32-bit	Valid, bus error negated
11	Reserved	Error, bus error asserted

Table 9.1 GPIO B_ERROR signal

9.2 Initialisation

On power-up, the three bus signals B_MODE[2:0] assume a status of (0,0,0). Also of interest are states of INI (0,1,0) and RST (0,1,1) which denote a bus-wide initialisation request and a soft reset state respectively. In case of any of these three bus conditions, the module will:

i) asynchronously initialise with a direction of 'IN' on all bits of the DIRECTION REGISTER (which will tristate the I/O pins);

ii) set a value of '0' on all bits of both the INPUT and OUTPUT Registers.

iii) Tri-state the BμILD Bus signal drivers.

All other B_MODE states are ignored (normal run state assumed).

9.3 GPIO Registers

The GP4020 GPIO interface has a Base Address of 0xE000 5000.

Address Offset	Register	Direction	Function
0x000	Direction	Read / Write	1 = input, 0 = Output
0x004	Input	Read	Data from pins
0x008	Output	Write	Data to pins if Direction = Output

Table 9.2 GPIO Register Map

note: Any read or write access to the unused address, or writes to Input Register, or reads from Output Register will cause B_ERROR to be asserted high.

All registers are addressable as 32-bit locations only.

9.3.1 GPIO Direction Register – GPIO_DIR - Memory Offset 0x000

A write to this port enables or disables the corresponding data bit driver to the external pin. Logic '1' disconnects the pin as in a high-impedance ('Z') state, which then acts as an input. A logic '0' enables drive to the pin, which then behaves as an output.

Bit No.	Mnemonic	Description	Reset Value	R/W
31:8	-	*Reserved*	-	-
7:0	*GPIO_DIR[7:0]*	GPIO Pin Direction. Each GPIO_DIR bit maps to a single GPIO signal. (i.e. GPIO_DIR[4] maps to GPIO[4] - (pin 95 (100-pin package))) '1' configures the pin as an input, '0' configures the pin as an output.	0xFF	R/W

Table 9.3 GPIO_DIR Register

9.3.2 GPIO Input Register – GPIO_INPUT - Memory Offset 0x004

Readable only, a write to this address will produce a b_error. The instantaneous voltage condition on the external pin is latched on each rising edge of ip_rd, which is synchronous with b_clk; and is available by a read from this address. Any Output Register bits configured as OUT will have the respective data values echoed back via the I/O pin, else the IN bits will read the external device.

In addition, if the echoed back value of an OUT-configured bit is different to the data value, this is evidence of an error, possibly an external output-driver clash.

Bit No.	Mnemonic	Description	Reset Value	R/W
31: 8	-	*Reserved*	-	-
7:0	*GPIO_INPUT[7:0]*	GPIO Input. Each GPIO_INPUT bit maps to a single GPIO pin and allows the value of the pin to be read. (i.e. GPIO_INPUT[4] maps to GPIO[4] (pin 95 (100-pin package)))	-	R

Table 9.4 GPIO_INPUT Register

9.3.3 GPIO Output Register – GPIO_OUTPUT - Memory Offset 0x008

Writeable only, a read to this address will produce a b_error. A write to this register stores logic values which are driven to the pins when enabled by an OUT in the corresponding DIRECTION REGISTER bit. When direction is IN the logic value will have no effect, although it is still stored in the latch.

Bit No.	Mnemonic	Description	Reset Value	R/W
31: 8	-	*Reserved*	-	-
7:0	*GPIO_OUTPUT*	GPIO Output. Each GPIO_OUTPUT bit maps to a single GPIO pin. (i.e. GPIO_OUTPUT[4] maps to GPIO[4] (pin 95 (100-pin package))). When configured as an output, a "1" drives a High output, a "0" drives a Low.	0x00	W

Table 9.5 GPIO_OUTPUT Register

10 INTERRUPT CONTROLLER (INTC)

The Interrupt Controller can manage upto 32 Interrupt sources. In the GP4020, 18 interrupt sources are present: 16 internal sources, and 2 external sources. The Interrupt controller processes these raw interrupt sources down into two main CPU interrupts. These are called FIQ and IRQ. The names come from the two prioritised interrupts on the ARM family of processors. The FIQ stands for “Fast Interrupt Request”, whereas IRQ is a regular interrupt request.

The differences between the two interrupt types are as follows:

1) FIQ interrupts have higher priority than IRQ interrupts. As such, an IRQ interrupt can be interrupted by an FIQ interrupt. However, an FIQ interrupt may not be interrupted by another interrupt. Also note that an IRQ interrupt cannot be interrupted by another IRQ interrupt.

2) There are “private” banked registers for use when the processor is in either FIQ or IRQ modes. In IRQ mode, there are two such registers. In FIQ mode, there are seven banked registers for the programmer’s use. This larger number of banked registers means that latency-sensitive interrupts can be processed without the overhead of having to save the context of the machine by stacking registers before use.

The Interrupt Controller is designed to work with the BμILD bus, and its control is via a series of registers from that bus. It can basically be considered as two large OR gates with a certain amount of pre-processing carried out on each channel before they are ORed together. Each interrupt channel has a hierarchical priority in terms of IRQ: Channel 0 has highest priority and channel 31 the lowest. FIQ has ultimate priority over all IRQ.

ADDRESS	EXCEPTION
0x00000000	Reset
0x00000004	Undefined Instruction
0x00000008	Software Interrupt
0x0000000C	Abort (prefetch)
0x00000010	Abort (data)
0x00000014	Reserved
0x00000018	IRQ
0x0000001C	FIQ

Table 10.1 Interrupt Vector Summary

Table 10.1 above shows the vector address associated with the various interrupts. The addresses shown are byte addresses, and will normally contain a branch instruction pointing to the relevant routine. The FIQ routine may reside at 0x1C onwards, thereby avoiding the need for (and execution time of) a branch instruction.

Each Interrupt channel can be separately configured to provide the following functions:

- Generation of either an FIQ or IRQ interrupt
- Independent masking of the channel interrupt source
- Status bits to indicate the state of the channel both before and after the masking
- Responds to edge-triggered or level-sensitive sources
- Programmable for active low or high interrupts
- A FIQ interrupt can be downgraded to an IRQ interrupt
- An IRQ interrupt can be upgraded to an FIQ interrupt

10: Interrupt Controller

In the GP4020, the interrupt channels are allocated as shown in *Table 10.2 below*. In each case the application software for the GPS receiver will need to configure the interrupt channels as shown.

The GP4020 Interrupt Controller has a Base Address of 0xE000 6000.

Further details for the programming of the Interrupt Controller can be found in *Section 5 of the "Firefly MF1 Core Design Manual" DM5003*, available from Zarlink Semiconductor.

Interrupt Channel	Interrupt Source	Function	Level/Edge Trigger
0	Watchdog	Watchdog approaching time-out. Watchdog requires reset key = 0xECD9F7BD	Level - Act HI
1	Correlator	ACCUM_INT - interrupt to service GPS Accumulation data	Level - Act HI
2	Correlator	MEAS_INT - interrupt to service GPS Measurement data	Level - Act HI
3	DMAC	Channel status change	Level - Act HI
4	Peripheral Control Logic	Interrupt to service peripheral function. Function to be serviced determined by read of PER_STAT register in PCL.	Level - Act HI
5	N/A	NOT USED. Tied to "0"	N/A
6	TIC1	TIC source 1 Time out A	Edge
7	TIC1	TIC source 1 Time out B	Edge
8	RF_PLL_LOCK	RF Front-end PLL is NOT locked	Level - Act LO
9 to 12	N/A	(NOT USED. Tied to "0")	N/A
13	BSIO	Serial Input / Output block interrupt	Level - Act HI
14	UART1	Transmit / Receive error or modem status change	Level - Act HI
15	UART1	Receive Buffer Full	Level - Act HI
16	UART1	Transmit Buffer Empty	Level - Act HI
17	N/A	(NOT USED. Tied to "0")"	N/A
18	TIC2	TIC source 2 Time out A	Edge
19	TIC2	TIC source 2 Time out B	Edge
20	IEXTINT2	External Interrupt needs service	User defined
21 to 25	N/A	(NOT USED. Tied to "0")	N/A
26	UART2	Transmit / Receive error or modem status change	Level - Act HI
27	UART2	Receive Buffer Full	Level - Act HI
28	UART2	Transmit Buffer Empty	Level - Act HI
29 to 31	N/A	(NOT USED. Tied to "0")	N/A

Table 10.2 GP4020 Interrupt Sources

11 MEMORY PERIPHERAL CONTROLLER (MPC)

11.1 Introduction

The Memory Peripheral Controller (MPC) is the interface between the BμILD bus and the external bus system. The MPC is a BμILD bus slave, which performs byte packing and sub memory width writes that allows bus masters to access external components of different widths, and use variable length read- and write wait-states. This allows the ARM to connect to nearly any memory component or parallel peripherals needed. The memory interface consists of a 20-bit address and 16-bit data bus along with the associated control signals.

Pin Name	Direction	Function
NSCS[0}	Output	Internal & External Boot ROM. Active low.
NSCS[1]	Output	External Chip Select 1. Active low.
NSCS[2A]	Output	External Chip Select 2. Active low.
NSWE[1:0]	Output	Write Enable, active low. Two provided for use with two 8-bit devices. One used with a single 8- or 16-bit device.
NSOE	Output	Output Enable, active low.
SADD[18:0]	Output	Address Bus
SDATA[15:0]	Input / Output	Data Bus
SWAIT	Input	Input, active High. Wait signal. Inserts wait states into current cycle.
NSUB	Output	Output, active Low. Selects Upper byte for Byte Writes to 16-bit SRAM. SADD[0] selects Lower Byte, and may be used as such for 8-bit SRAM.

Table 11.1 External Memory Bus Pinout

The control of the MPC involves the use of four areas of Configuration, which are covered by four internal select lines: NCS[0], NCS[1], NCS[2] and NCS[3]. Each of these is configured to drive a separate area of memory in a GP4020 based system.

The mapping of the NCS lines to the NSCS lines is not direct, as the data in *Table 11.2 below* shows.

Address Offset	Register	Firefly MF1 Core select	External Chip Select	Function
0x000	Area 1 Config.	NCS[0]	NSCS[0]	Controls Internal / External Boot ROM Memory
0x004	Area 2 Config.	NCS[1]	NSCS[1]	Controls External SRAM memory chip 1
0x008	Area 3 Config.	NCS[2]	NSCS[2A]	Controls External SRAM memory chip 2A and some internal components: 12-channel Correlator, 1PPS Timemark Generator, System Clock Generator, Real Time Clock and Peripheral Control Logic.
0x00C	Area 4 Config.	NCS[3]	N/A	Controls Internal SRAM, (2k x 32-bit)

Table 11.2 Memory Peripheral Controller Configuration Registers

The MPC Configuration registers are used to configure the bus-width, the wait-state timing, and the read-only state of each memory area controlled by the MPC.

11.2 GP4020 Memory Area 1 Configuration

GP4020 Memory Area 1 is at the base of the Address Map (0x0000 0000 through to 0x000F FFFF), and is typically shared between an internal BOOT ROM and an external 16-bit FLASH EPROM via chip-select line NSCS[0]. The control of whether the BOOT ROM or the External memory space is selected, is controlled by the state of MULTI_FNIO (pin 54 (100-pin package)) at chip reset. Refer to *Section 4 "BOOT ROM" on page 27* for more information.

The default hard-wired configuration at Reset of MPC Memory Area 1 (register address 0xE000 8000) is 0xFF00 0035:

Access Waits	bits [31:28]	= '0y1111'	= 15 wait states
Stop Waits	bits [27:24]	= '0y1111'	= 15 wait states
Reserved	bits [23:8]	= '0x0000'	
Configuration Mode	bits [7:6]	= '0y00'	(Mode 0 – See Note 1)
Wait Control	bit [5]	= '1'	(enable MPC control of Wait-states)
Read Only Status	bit [4]	= '1'	(Read Only enabled)
Sub Memory Write Status	bit [3]	= '0'	(Sub memory writes disabled)
Access Type	bit [2]	= '1'	(Memory access – See Note 2)
Data Size	bits [1:0]	= '0y01'	(Half Word (16-bit) wide)

Notes:

1) The Configuration Mode bits set the MPC Configuration for Area 1 to "Butterfly Mode" (Mode 0) by default. Butterfly Mode does NOT give the same level of configuration flexibility as Standard Mode (Mode 1). (Refer to details for this in the *Memory Peripheral Controller section (Section 3) of the "Firefly MF1 Core Design Manual" DM5003*, available from Zarlink Semiconductor.)
2) Memory Access Type: signifies that a transaction may be constructed from multiple memory accesses; i.e. operand size does not have to equal memory size.
3) '0x0000' signifies a HEX number value of '0000'; '0y0000' signifies a BINARY number value of '0000'.

With the default settings shown above, a 16-bit ROM can be read, with 15 wait-states. One of the first things to be done in software after boot will be to configure the MPC Configuration for Area 1 to more optimal settings. Typically, where a FLASH EPROM is used with NSCS[0], the access time tends to be quite slow (70ns to 100ns) but NOT slow enough to need 15 wait-states. If the BμILD_CLK for the Firefly MF1 is set to 20MHz (default value), then the period of each clock-cycle is 50ns. Therefore to access a 100ns FLASH EPROM, 2 access wait-states will need to be introduced to guarantee that the data addressed in memory returns to the Firefly in time.

11.3 GP4020 Memory Area 2 Configuration

GP4020 Memory Area 2 is specified by addresses 0x2000 0000 through to 0x200F FFFF, and is typically used by high-speed external 16-bit SRAM via chip-select line NSCS[1].

The default hard-wired configuration at Reset of MPC Memory Area 2 (register address 0xE000 8004) is 0xFF00 0034:

Access Waits	bits [31:28]	= '0y1111'	= 15 wait states
Stop Waits	bits [27:24]	= '0y1111'	= 15 wait states
Reserved	bits [23:8]	= '0x0000'	
Configuration Mode	bits [7:6]	= '0y00'	(Mode 0 configuration)
Wait Control	bit [5]	= '1'	(enable MPC control of Wait-states)
Read Only Status	bit [4]	= '1'	(Read Only enabled)
Sub Memory Write Status	bit [3]	= '0'	(Sub memory writes disabled)
Access Type	bit [2]	= '1'	(Memory access)
Data Size	bits [1:0]	= '0y00'	(Byte (8-bit) wide)

With the default settings shown above, an 8-bit SRAM can be read (but NOT written to), with 15 wait-states. One of the first things to be done in software after boot will be to configure the MPC Configuration for Area 2 to more optimal settings. Typically, where a SRAM is used with NSCS[1], the access time tends to be fast (10ns to 20ns). If the BμILD_CLK for the Firefly MF1 is set to 20MHz (default value), then the period of each clock-cycle is 50ns. Therefore, to access a 10ns FLASH EPROM, no access wait-states are required.

11.4 GP4020 Memory Area 3 Configuration

GP4020 Memory Area 3 is a special case where a number of internal components share resources with an External chip select line – NSCS[2A]. The 12-channel correlator which resides in Memory Area 3 uses a re-timing UIM interface, to retime data from Firefly speed (any frequency from the System Clock Generator (SCG)) to the Correlator speed (nominally 40MHz).

The default hard-wired configuration at Reset of MPC Memory Area 3 (register address 0xE0008008) is 0xFF000034:

Access Waits	bits [31:28]	= '0y1111'	= 15 wait states
Stop Waits	bits [27:24]	= '0y1111'	= 15 wait states
Reserved	*bits [23:8]*	*= '0x0000'*	
Configuration Mode	bits [7:6]	= '0y00'	(Mode 0 configuration)
Wait Control	bit [5]	= '1'	(enable MPC control of Wait-states)
Read Only Status	bit [4]	= '1'	(Read Only enabled)
Sub Memory Write Status	bit [3]	= '0'	(Sub memory writes disabled)
Access Type	bit [2]	= '1'	(Memory access)
Data Size	bits [1:0]	= '0y00'	(Byte (8-bit) wide)

As memory area 3 is used to access either the 12-channel correlator, Internal Peripheral Functions or NSCS[2A] external chip-select line, the settings of the MPC must reflect the memory configuration that is being used.

The dominant speed constraint in Memory Area 3 is the 12-channel correlator, which must have a 175ns access-time specified in the configuration register. With a 20MHz firefly clock (50ns period) this equates to a minimum 4 wait-states in the Firefly clock.

In order to avoid complex re-configuring of the memory 3 area during code execution, it is recommended that any external memory component used in Memory Area 3, should be slow-speed - 100ns access or slower is recommended.

When the MPC changes its addressing from External to Internal destinations in Memory Area 3, there will be an access delay at the point of first access of 1-wait-state, due to the way the data-bus I/Os are configured. However, further accesses to the same part of the address space will NOT incur any further access delays apart from those derived from accessing the correlator through the UIM interface. It is consequently recommended that "Start Write Waits" and "Start Read Waits" be introduced when any externally accesses are required for Memory Area 3.

In order to guarantee reliable access to both external and internal memory components in Memory Area 3 when the Firefly is being clocked at 20MHz or higher, it is recommended that a default of 3 Start-Wait States and 3 Access-Wait-states are programmed into the MPC Area 3 Configuration register.

Consequently, it is strongly recommended that after boot-up of any system software, the MPC Configuration register for Area 3 memory is configured as follows:

Start Write Waits	bits [31:28]	= '0y0011'	= 3 Start Waits
Access Write Waits	bits [27:24]	= '0y0011'	= 3 Access Waits
Stop Write Waits	bits [23:20]	= '0y0000'	
Start Read Waits	bits [19:16]	= '0y0011'	= 3 Start Waits
Access Read Waits	bits [15:12]	= '0y0011'	= 3 Access Waits
Stop Read Waits	bits [11: 8]	= '0y0000'	
Configuration	bits [7:6]	= '0y01'	(Mode 1 – Standard Mode)
Wait Control	bit [5]	= '1'	(enable MPC control of Wait-states)
Read Only Status	bit [4]	= '0'	(Read Only NOT enabled)
Access Sub Memory Type	bits [3:2]	= '0y11'	(allows 8, 16 & 32-bit wide access)
Data Size	bits [1:0]	= '0y10'	(32-bit wide)

Essentially, this equates to setting address 0xE000 8008 to a value of 0x3303 306E.

The MPC must be configured to address a 32-bit bus when accessing the Area 3 internal peripherals.

By using “Sub-memory access”, writes to the 12-channel correlator can be either 16-bit or 32-bit; 8-bit accesses are NOT permitted. In the case of a 32-bit Write access, the 16MSBs of each access are ignored. In the case of a 32-bit read access, the 16 MSBs of the data will be set to ‘0x0000’.

Writes to all other parts in Area 3 can be 8-, 16-, or 32-bit wide.

11.5 GP4020 Memory Area 4 Configuration

GP4020 Memory Area 4 is used to access the internal SRAM only in the GP4020, and NO other components. The internal SRAM is configured as 2k x 32-bit data or 8k bytes and can be configured to have either 8-bit, 16-bit or 32-bit data width. Also, the access time of the internal SRAM is high speed; only 6ns, and so can be accessed with 0-wait-states for any Firefly MF1 BμILD CLK frequency setting.

The default hard-wired configuration at Reset of MPC Memory Area 4 (register address 0xE000800C) is 0xFF000034:

Access Waits	bits [31:28]	= ‘0y1111’	= 15 wait states
Stop Waits	bits [27:24]	= ‘0y1111’	= 15 wait states
Reserved	*bits [23:8]*	*= ‘0x0000’*	
Configuration Mode	bits [7:6]	= ‘0y00’	(Mode 0 configuration)
Wait Control	bit [5]	= ‘1’	(enable MPC control of Wait-states)
Read Only Status	bit [4]	= '1'	(Read Only enabled)
Sub Memory Write Status	bit [3]	= ‘0’	(Sub memory writes disabled)
Access Type	bit [2]	= ‘1’	(Memory access)
Data Size	bits [1:0]	= '0y00'	(Byte (8-bit) wide)

With the default settings shown above, an 8-bit SRAM can be read (but NOT written to), with 15 wait-states. One of the first things to be done in software after boot will be to configure the MPC Configuration for Area 4 to more optimal settings. As the internal SRAM is high-speed, 0-wait-state access can be used across any data width.

ALL the MPC registers are accessible at System Base Address 0xE000 8000.

Further details for the programming of the Memory Peripheral Controller can be found *in Section 3 of the "Firefly MF1 Core Design Manual" DM5003*, available from Zarlink Semiconductor.

12 PERIPHERAL CONTROL LOGIC (PCL)

12.1 Introduction

The Peripheral Control Logic (PCL) is used to control GP4020 chip-wide functions. The PCL can be considered to have the following discrete functions:

1) Chip Reset logic
2) PLL Enable logic
3) Multiplex logic
4) Interrupt and wake-up logic
5) Chip status monitoring.
6) Chip-wide power-control
7) Test mode set-up

Figure 12.1 below shows a block diagram of most of the functions that the PCL performs. Note that Chip-wide power-control and Test-mode set-up have not been explicitly indicated in the diagram.

12.2 Chip Reset Logic

The GP4020 uses a number of sources to allow chip-wide resetting of clocks and registers. *Figure 12.2 on page 115* shows the logic contained within the Reset Logic Block of the PCL. The main sources for reset come from the following signals:

- **RF_PLL_LOCK** (Pin 56 (100-pin package)) – when Low, this indicates that the RF Front-end PLL is not locked. Therefore the clocks to & from the RF Front-end (CLK_T (Pin 58), CLK_I (Pin 59), SAMPCLK (Pin 63)), and the digitised IF on the SIGN0 and MAG0 inputs (pins 61 and 62), are NOT valid. A reset of the Firefly MF1 and the 12-channel correlator will occur as shown in *Figure 12.3 on page 116*, if RF_PLL_LOCK is taken Low (i.e. '0'). In addition, this line feeds the Interrupt Controller (INTC), which can respond to this if configured to do so.

  The RF_PLL_LOCK input can be inhibited from resetting the Firefly and Correlator by ensuring that the EN_PLL_RST bit of the “PER_STAT” register is set to “0”.

- **POWER_GOOD** (Pin 64 (100-pin package)) – when Low, this indicates that the voltage supply to the GPS receiver is below a limit defined by a Power-on reset reference comparator on the RF Front-end. A reset of the Firefly MF1 and the 12-channel correlator will occur as shown *in Figure 12.4 on page 116*. In this diagram, it is assumed that a clock is present throughout the POWER_GOOD event.

  The POWER_GOOD input can be inhibited from resetting the Firefly and Correlator by ensuring that the EN_POW_RST bit of the “PER_STAT” register is set to “0”.

  In a GPS receiver using the GP2010 or GP2015 RF Front-end, a POWER_GOOD event normally signifies the loss of power to the RF Front-end also. This results in RF_PLL_LOCK being set Low, and the disappearance of CLK_T and CLK_I clock signals. When Power is re-applied, the POWER_GOOD line will be set High quickly after power-up. However, if a GP2010 or GP2015 RF Front-end is used, the PLL in the RF chip will typically take 5ms from power-up to set RF_PLL_LOCK to High. This is shown in *Figure 12.5 POWER_GOOD Hardware Reset Generation when POWG_EN = '1'.* Assumes that power to RF Front-end fails, and RF_PLL_LOCK is low for upto 5ms after power-up. *on page 116*.

  Also, if POWG_EN (bit 15 in the POW_CNTL register) is set to '1', the POWER_GOOD line will also power down the whole GP4020 chip, except for the Real Time Clock and Data Retention register. A reset of the Firefly MF1 and the 12-channel correlator will occur, along with a power-down of most GP4020 functions. Refer to *Section 12.6 "Chip-wide Power Control modes" on page 123*, for more information.

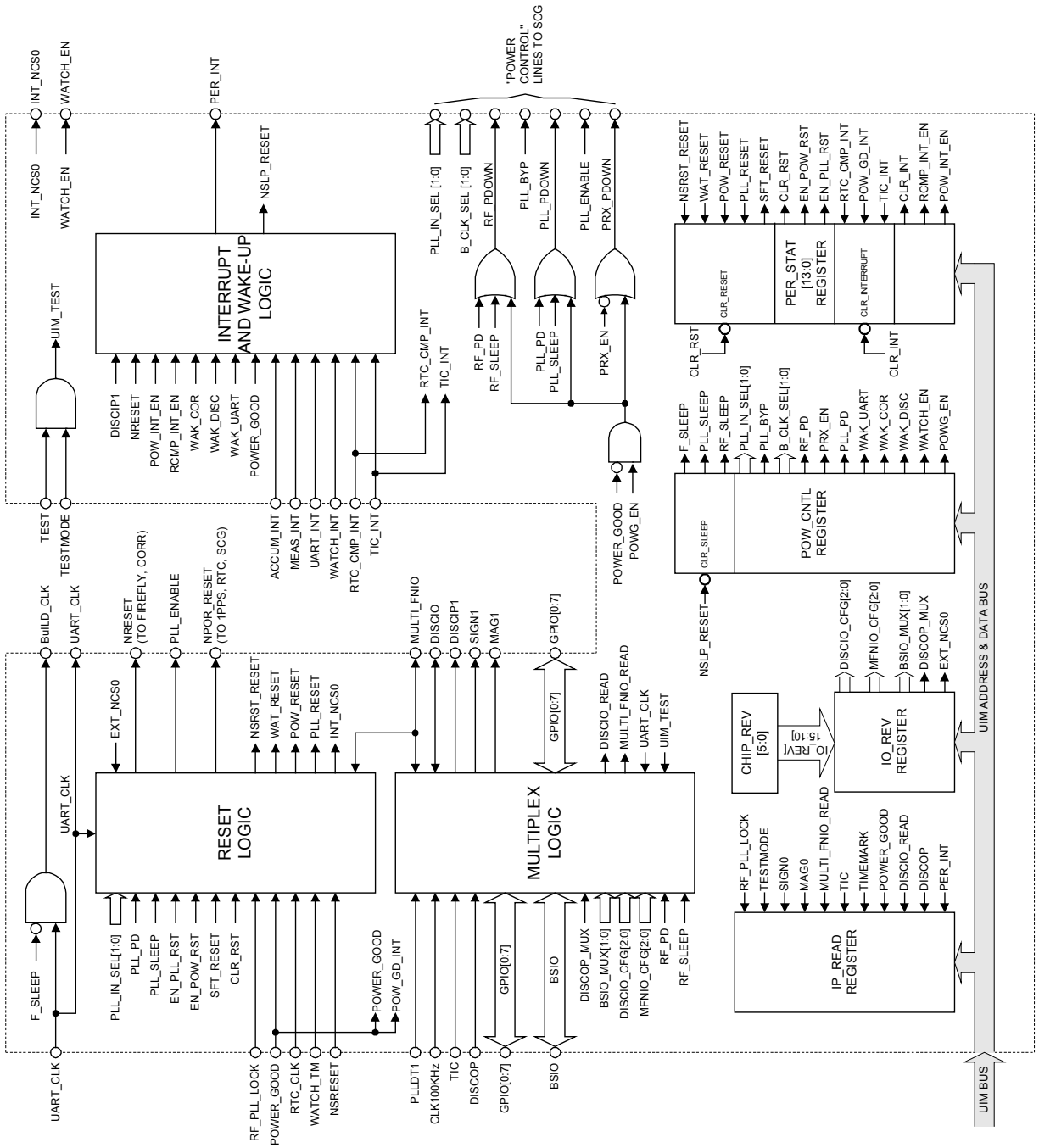


Figure 12.1 Peripheral Control Logic Top-level Block Diagram

- **NSRESET** (Pin 75 (100-pin package)). This indicates that a RESET has been requested from an external source, perhaps a Reset Button. A system reset will occur as shown in *Figure 12.6 on page 117*, if this pin is taken Low (i.e. '0'). This is the only reset source by which ALL GP4020 registers, which can be reset, are completely reset (except for the Data Retention Register in TIC_RET within the 1PPS Timemark Generator, and the Real Time Clock Counters). An active Low pulse of greater than 10ns is required on this pin in order to guarantee that a reset occurs. The NSRESET pin is the only hardware-reset source that can reset the PER_STAT [4:0] register bits, used to indicate sources of reset.

- **WATCH_TM** (or Watchdog Time-out) signal. This is an internally generated Reset signal comes from the on-chip Watchdog module. This will occur if the Watchdog has interrupted the Firefly MF1, but an interrupt service-routine in software has failed to reset the Watchdog. The Watchdog is reset by sending a 32-bit reset key within approx. 200µs (refer to *Section 18 "WATCHDOG TIMER (WDOG)" on page 176*, for more information). A system reset will occur as shown in *Figure 12.7 on page 117*. The WATCH_TM reset source will reset all registers which NSRESET will reset, except for PER_STAT [4:0] register bits, which are only reset by an NSRESET event.

- A software Reset signal, **SFT_RESET**, can be set to initiate a reset of the Firefly MF1 and the 12-channel correlator by writing a '0' to bit 4 of the PER_STAT register. The SFT_RESET will then cycle low for one UART_CLK cycle. This could be activated in conjunction with a software interrupt service routine and the RF_PLL_LOCK interrupt signal into the INTC block, but could also undertake a reset in some other software circumstances. The reset occurs as shown in *Figure 12.8 on page 117*. The reset state for this signal is '0' for a read and '1' for a write.

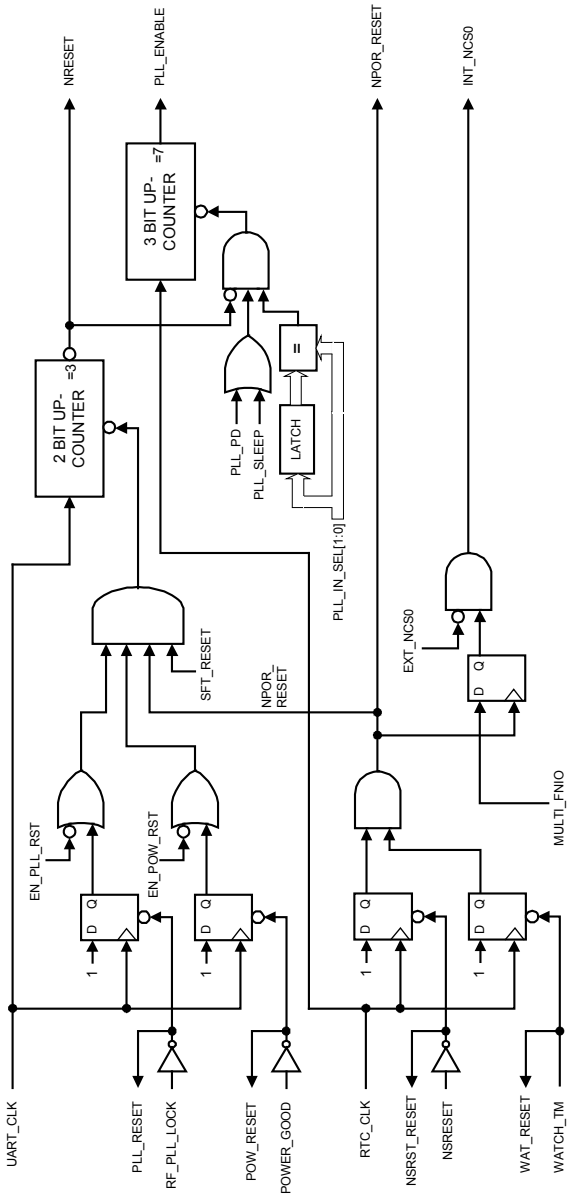


Figure 12.2 Peripheral Control Logic - Reset Logic

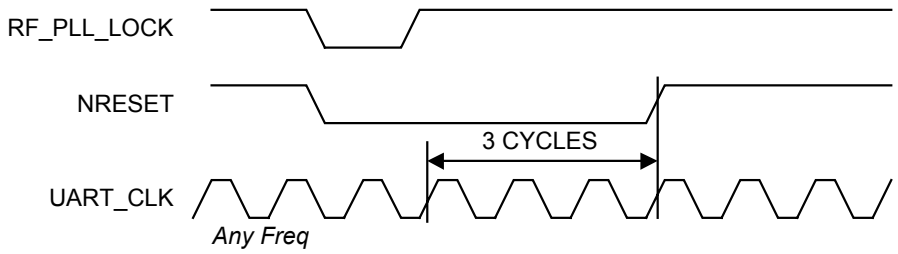


Figure 12.3 RF_PLL_LOCK Hardware Reset Generation

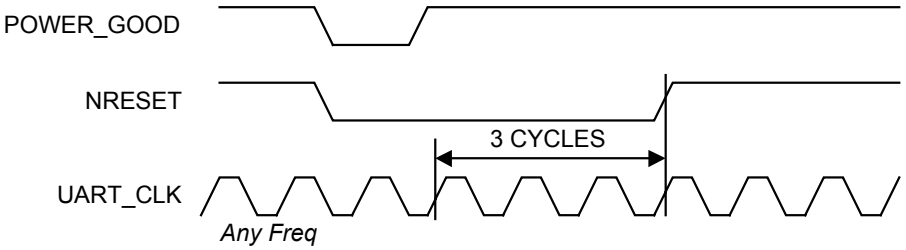


Figure 12.4 POWER_GOOD Hardware Reset Generation when POWG_EN = '0', and UART_CLK NOT derived from an RF Front-end.

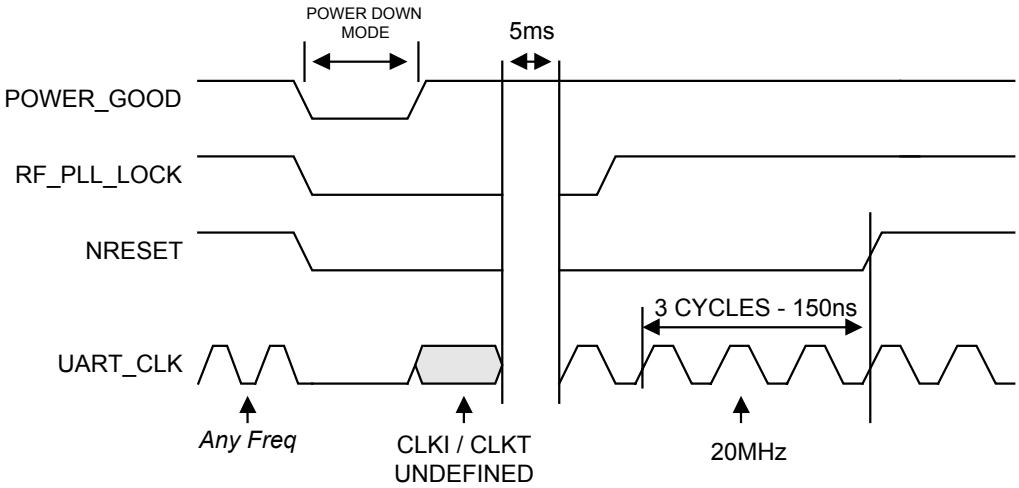
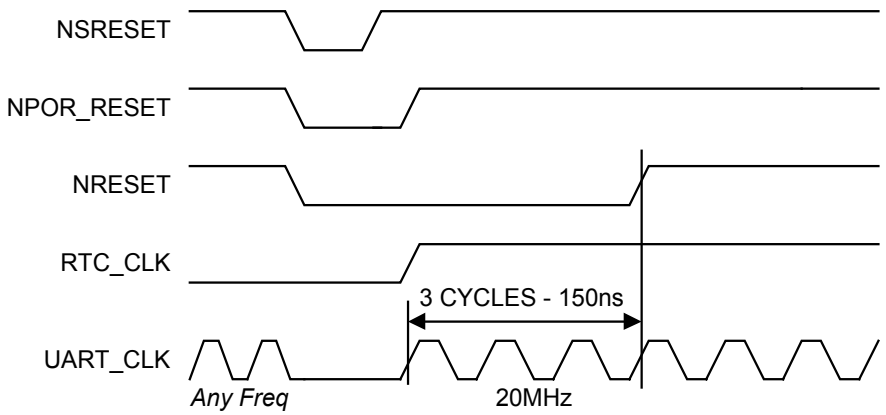
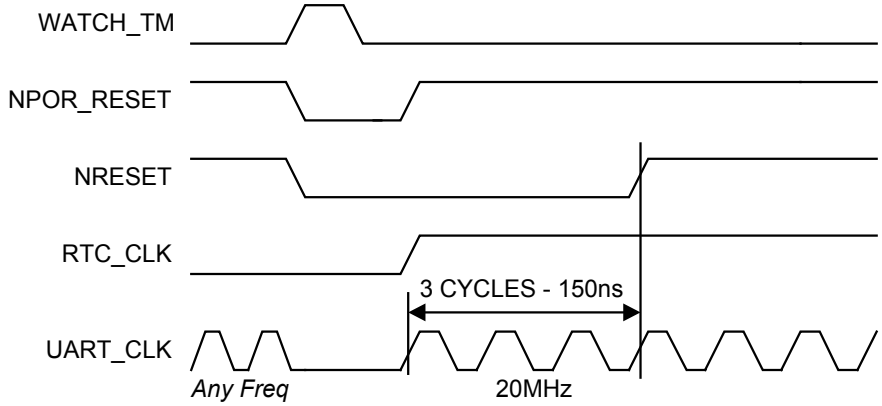


Figure 12.5 POWER_GOOD Hardware Reset Generation when POWG_EN = '1'. Assumes that power to RF Front-end fails, and RF_PLL_LOCK is low for upto 5ms after power-up.



RTC_CLK period = 30517ns. Rising edge only shown.

Figure 12.6 NSRESET Hardware Reset Generation



RTC_CLK period = 30517ns. Rising edge only shown.

Figure 12.7 Watchdog Hardware Reset Generation

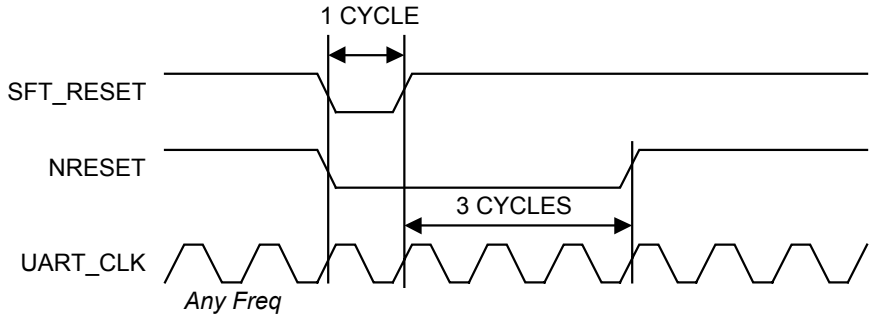


Figure 12.8 SFT_RESET Hardware Reset Generation

As shown in the Reset Logic circuit in *Figure 12.2 on page 115*, there are two internal reset lines produced in the GP4020 from the Reset inputs:

1) **NPOR_RESET.** Used to reset all registers in the Peripheral Control Logic, System Clock Generator and 1PPS Timemark Generator, and some registers in the Real Time Clock (see below for exceptions). It is derived from

either WATCH_TM or NSRESET signals. Therefore, a complete GP4020 reset can only occur if an NSRESET or a WATCH_TM event is introduced. These are synchronised to the 32kHz clock developed by the Real Time Clock, and the resulting output is active low. Additionally, the Reset status bits in PER_STAT[4:0] will only be reset by an asynchronous NSRESET event or a set of the CLR_RST bit to '0' in PER_STAT register, and will NOT be cleared by any other reset. An activation of NPOR_RESET will disable any System Clock settings using the PLL, and consequently, the System Clock will return to 20MHz (M_CLK / 2) when the reset is released.

2) **NRESET**. Used to reset the 12-channel correlator, the Firefly MF1 and all BμILD bus modules. It is derived from either: NPOR_RESET, RF_PLL_LOCK and POWER_GOOD inputs, or a Software Reset signal SFT_RESET. This reset line is activated for every type of reset input, and is synchronised to the Firefly Clock (UART_CLOCK), in conjunction with a 3-cycle delay. An activation of NRESET independent of NPOR_RESET will NOT disable the System Clock settings using the PLL, but will inhibit the BμILD CLK to the Firefly MF1 Core, and hence power down the Firefly MF1.

There are three areas of the GP4020 where a hard reset resulting in either an NPOR_RESET or NRESET signal, has no effect:

1) Real Time Clock counters:

   a) RTC_PRE[15:1] in the RTC_PRE register

   b) All data bits in register RTC_SEC_B

   c) RTC_SEC_T[7:0] in the COMPS_RTCS register

   The only way the Real Time Clock Counters can be reset is to write a '0' to bit 0 of the RTC_PRE register (refer to *Section 13 "REAL TIME CLOCK (RTC)" on page 131*, for more information).

2) Data Retention Register in 1PPS Timemark Generator (TIC_RET[15:8]). There is no asynchronous reset signal for the Data Retention Register.

3) Internal SRAM (2k x 32-bit).

12.3 PLL Enable Logic

An enable line (PLL_ENABLE) is used to re-enable the PLL block in the System Clock Generator after a reset, due to:

1) PLL Power-down, due to PLL_PD being active and then cleared.

2) PLL Sleep, due to PLL_SLEEP being active and then cleared by a "Wake-up" event.

3) Master Reset due to NRESET being active.

4) Change of PLL frequency due to a change of input reference frequency, controlled by a new setting on PLL_IN_SEL[1:0].

PLL_ENABLE uses a delay of seven cycles of the Real Time Clock (upto 183μs) to ensure that the PLL clock output is only enabled after the PLL output frequency is stable on the GP4020 chip. The timing diagram in *Figure 12.9 below* illustrates this.

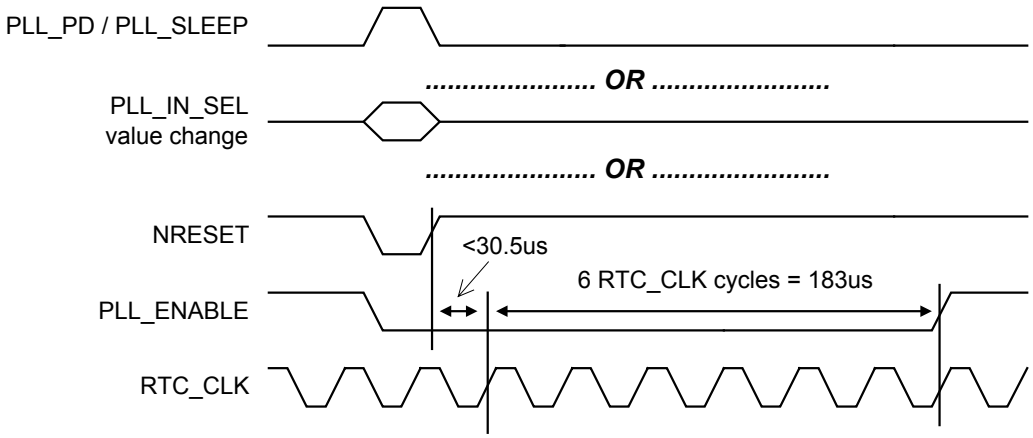


Figure 12.9 PLL_ENABLE Timing

The selection of an external ROM in place of the Internal Boot ROM can also be determined during a reset. The INT_NCS0 line will go High, and the Internal ROM will be selected, if EXT_NCS0 (IO_REV[9]) is set Low and MULTI_FNIO pin is held High, whilst the NPOR_RESET line toggles from Low to High during a reset. If MULTI_FNIO is held Low under the same scenario, the Internal ROM will NOT be selected after the reset, and the state of EXT_NCS0 will be ignored.

12.4 Multiplex Logic

The standard GP4020 is packaged in a 100-pin package. Ten additional signals can be accessed non-simultaneously via some configurable Input / Output lines, as shown in *Figure 12.10 below*. The GP4020 pins that allow this are:

1) **DISCIO** (pin 55 (100-pin package)). This can be configured using the DISCIO_CFG[2:0] control lines from the IO_REV register as shown in *Table 12.1 on page 120*.

2) **MULTI_FNIO** (pin 54 (100-pin package)). This can be configured using the MFNIO_CFG[2:0] control lines, from the IO_REV register as shown in *Table 12.2 on page 120*.

3) **GPIO[7:0]** (pins 91, 92, 93, 95, 96, 97, 99, 100 (100-pin package)). These can be configured using the BSIO_MUX[1:0] and DISCOP_MUX control lines from the IO_REV register, and the UIM_TEST input control line. Primarily used to interface to General Purpose Input Output (GPIO) cells, but can be multiplexed to BSIO and other signals as shown in *Table 12.3 on page 121.*

All the pins indicated are tolerant to 5V input levels, so interfacing to an existing 5V-logic system is easy. The use of the DISCOP and DISCIP1 lines from the 12-channel correlator allows access for test signals during manufacturing test.

The DISCIP1 line can be also used as a discrete input, which can be read by the ACCUM_STATUS_B register in the 12-channel correlator block.

The DISCOP line can also be used as a multi-function output line, as configured by the SYSTEM_SETUP register in the 12-channel correlator block. It can be set to deliver either a CLK100KHz output from 12-channel correlator block, or the RAW_Timemark signal.

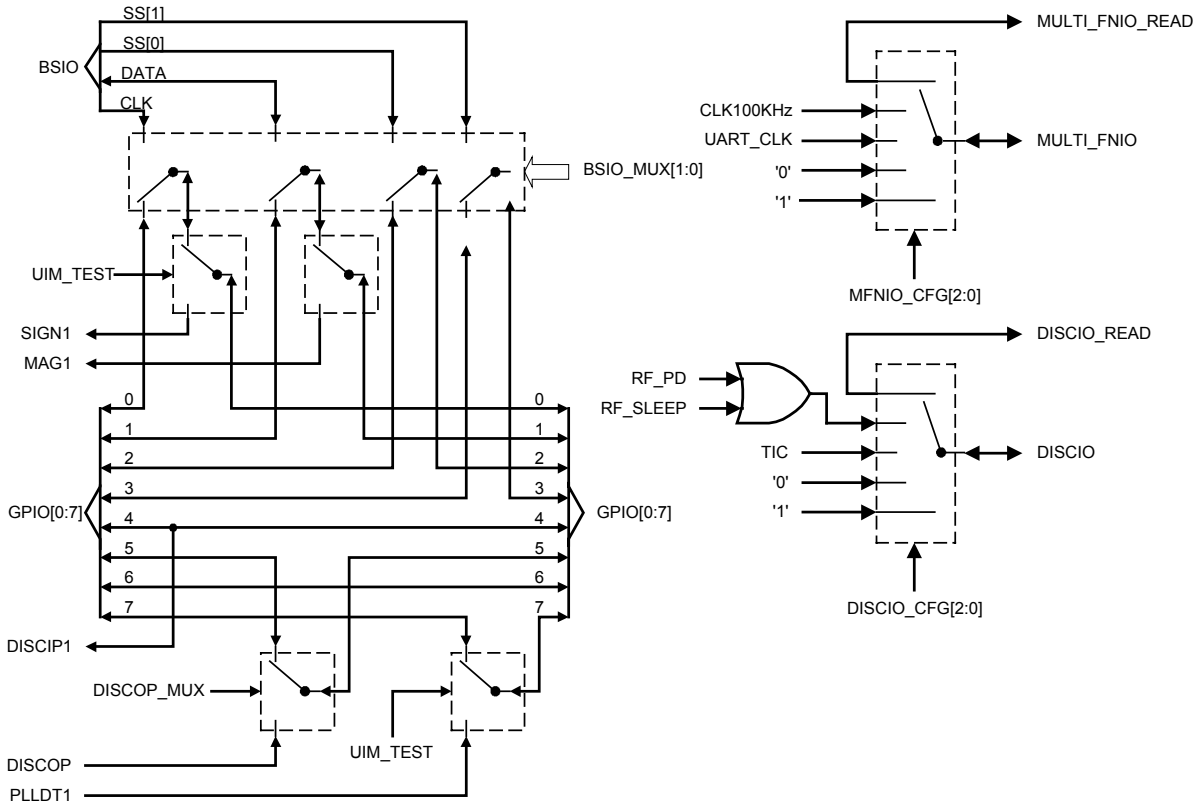


Figure 12.10 Peripheral Control Logic - Multiplex Logic

DISCIO_CFG[2:0]	DISCIO Function
0xx	Input only; read using DISCIO bit in IP_READ register. Also used by Correlator in UIM Test Mode.
100	RF_PD or RF_SLEEP output to RF Front-end. Note if this mode used, it is recommended that 1kohm pull-down resistor used on this pin, since reset-condition is to make DISCIO pin an input, which will mean that voltage on PDn input on RF Front-end will be >+0.8V, and could disable the RF Front-end IC.
101	TIC output
110	'0' output
111	'1' output

Table 12.1 DISCIO pin signal multiplex options

MFNIO_CFG[2:0]	MULTI_FNIO Function
0xx	Input only; read using MULT_FNIO bit in IP_READ register. Also used by Correlator in UIM Test Mode.
100	CLK100KHz output from 12-channel correlator block. This signal is a 100kHz square wave, phase-locked to RF Front-end PLL.
101	UART_CLK output.
110	'0' output
111	'1' output

Table 12.2 MULTI_FNIO pin signal multiplex options

GPIO output line number	Alternative signal multiplex	Condition
0	BSIOCLK	BSIO_MUX[1:0] = '10', '01', '11'
	SIGN 1 *Not available in standard operation*	UIM_TEST = '1' (i.e. TEST = High TESTMODE = High)
1	BSIODATA	BSIO_MUX[1:0] = '10', '01', '11'
	MAG 1 *Not available in standard operation*	UIM_TEST = '1' (i.e. TEST = High TESTMODE = High)
2	BSIOSS[0]	BSIO_MUX[1:0] = '01', '11'
3	BSIOSS[1]	BSIO_MUX[1:0] = '10', '11'
4	DISCIP1 input to Correlator	Always connected
5	DISCOP output from Correlator	DISCOP_MUX = '1'
6	N/A	
7	PLLDT1 output	UIM_TEST = '1'

Table 12.3 GPIO pin signal multiplex options

12.5 Interrupt and Wake-up logic

The Interrupt and Wake-up logic block of the Peripheral Control Logic block is used to create a single interrupt line to the Firefly MF1, from a number of interrupt sources. It also sets up the hardware for a number of Wake-up-from-Sleep events. The logic used to set-up these two facilities is shown in *Figure 12.11 below.*

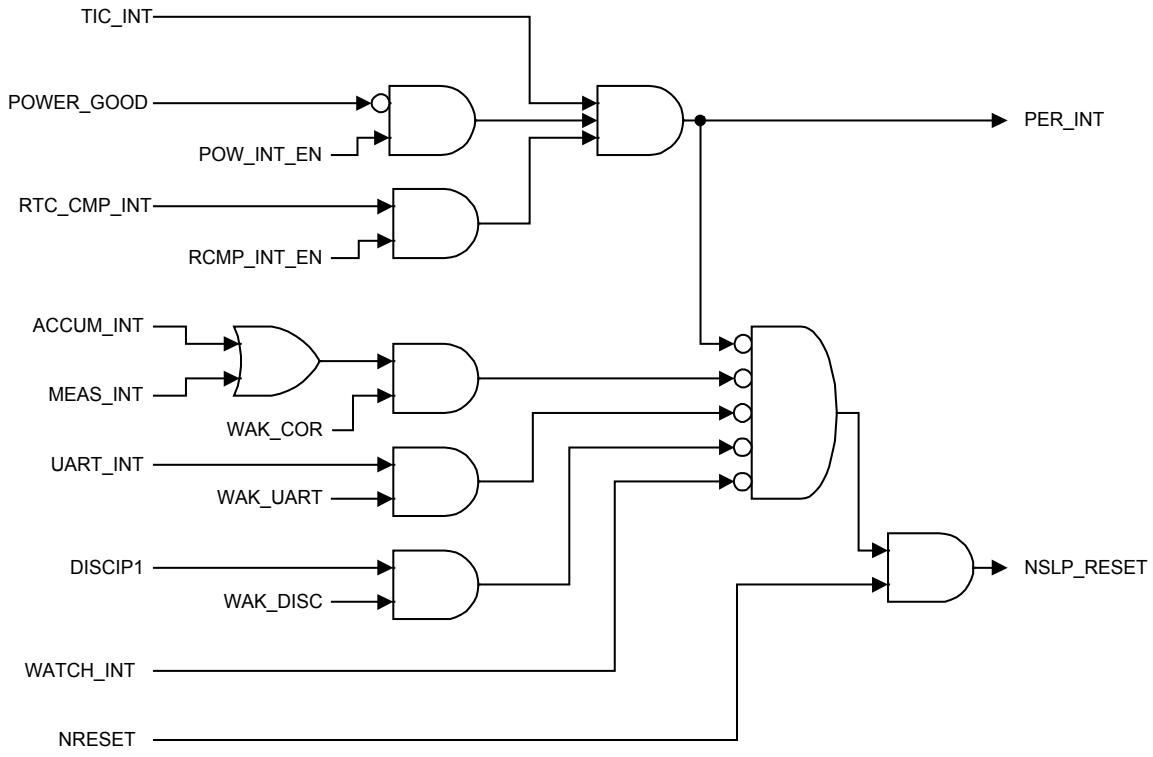


Figure 12.11 Peripheral Control Logic - Peripheral Interrupt and Wake-up control logic

12: Peripheral Control Logic

A single Interrupt line, PER_INT, is produced from 3 Peripheral Control Logic Interrupt signals from the Real Time Clock (RTC_CMP_INT), the 1PPS Timemark Generator (TIC_INT), and the POWER_GOOD input (Pin 64 (100-pin package)). The PER_INT interrupt is connected to the Firefly Interrupt Controller, and the signal which generated PER_INT interrupt can be determined from the RTC_CMP_INT, POW_GD_INT and TIC_INT bits of the PER_STAT register.

A wake-up event occurs when the NSLP_RESET internal reset line toggles from High to Low. GP4020 modules that have been put to sleep can be awoken by a wake-up event. There are three sleep enable bits in the POW_CNTL register, which are cleared by the NSLP_RESET signal at each Wake-up event:

1) **F_SLEEP.** When enabled, the BμILD Clock to the Firefly MF1 (and most of external BμILD bus modules) is inhibited. The Watchdog and UART2 are NOT disabled by this action; they are sourced with the UART_CLK signal, NOT BμILD Clock.

2) **PLL_SLEEP.** When enabled, the PLL in the System Clock Generator module is disabled.

3) **RF_SLEEP.** When enabled, the 40MHz Low Level Differential Input cell of the System Clock Generator is disabled. In addition, if the DISCIO pin (Pin 55 (100-pin package)) is set by the Multiplex Logic (see earlier) to connect to RF_SLEEP and RF_PD, the RF Front-end IC will also be disabled.

The Wake-up event can be created by any of the following scenarios:

a) **TIC_INT interrupt**, from the 1PPS Timemark Generator. The TIC_INT can be programmed to occur at every TIC; nominally once every 0.0999999s, or every time the 1PPS generator automatically extends a TIC period by 175ns. This feature can be enabled / inhibited using TIC_INT_EN[1:0] in the PER_STAT register.

b) **POWER_GOOD failure**. This allows the full GP4020 chip to be re-awaken if a Power-supply failure has been detected (only works effectively, if POWG_EN in POW_CNTL is set to '0'). This feature can be enabled / inhibited using POW_INT_EN in the PER_STAT register.

c) Real Time Clock interrupt (RTC_CMP_INT). The Real Time Clock can be set with a comparison value, which when the accumulated time reaches the comparison value, the RTC_CMP_INT line goes High. Essentially, this is a variable length wake-up timer facility, which allows blocks of the GP4020 to be put to sleep for any period from between 30.5us and 256s. This feature can be enabled / inhibited using RCMP_INT_EN in the PER_STAT register**.**

d) **Accumulated Data Interrupt (ACCUM_INT) or Measurement Data Interrupt (MEAS_INT)** from the 12-channel Correlator. An ACCUM_INT interrupt is produced each time the correlator requires the microprocessor to retrieve Accumulated GPS data from the Accumulation registers in each of the tracking modules. This is nominally once every 505μs but can be adjusted using the PROG_ACCUM_INT register in the 12-channel correlator to be between 175ns and 1.43ms. The MEAS_INT interrupt is a signal derived from the 12-channel correlator TIC signal, and is nominally produced 50ms before each TIC. This feature can be enabled / inhibited using WAK_COR in the POW_CNTL register.

e) **UART_INT interrupt**, caused by new external data being received by the UART2 Rx input (pin 77 (100-pin package)). This feature can be enabled / inhibited using WAK_UART in the POW_CNTL register.

f) **Discrete input interrupt (DISCIP1).** When a High is detected on GPIO[4] (pin 95 (100-pin package)), this can be used to produce a wake-up event. This feature can be enabled / inhibited using WAK_DISC in the POW_CNTL register.

g) **Watchdog interrupt (WATCH_INT)**, prior to a Watchdog time-out. The Watchdog produces two signals:

- WATCH_INT which is used to interrupt the microprocessor to instruct it to reset the watchdog;
- WATCH_TM, which is used to indicate that the Watchdog has interrupted the microprocessor for attention, but the microprocessor has NOT responded.

When a WATCH_INT signal is produced (period of which is programmable within the Watchdog block), this can be used to wake-up the microprocessor, if it is in sleep mode. The WATCH_INT wake-up event cannot be disabled unless the watchdog itself is disabled, by clearing the WATCH_EN bit of the POW_CNTL register.

Note: the WATCH_EN bit in only effects Watchdog behaviour due to Firefly reset. (If WATCH_EN is set to '1', the Watchdog starts immediately. If WATCH_EN is '0', Watchdog will only start after the RESTART KEY value is written to it).

h) Firefly and Correlator block reset, due to NRESET. This feature cannot be disabled.

12.6 Chip-wide Power Control modes

The GP4020 incorporates a number of disable and power-saving modes.

12.6.1 Full Power-Down

A Full Power-down of the GP4020 (i.e. Clocks removed from the whole chip, except for the Real Time Clock) can be made to occur, if a '1' is written to the POWG_EN bit (POW_CNTL[15]), and the POWER_GOOD input (pin 64 (100-pin package)) is set Low. Under these conditions, the following pins on the GP4020 get set as shown (pin numbers refer to 100-pin package):

- Set as High Impedance (i.e. Tristate):

  NSCS[0] (pin 11), NSCS[1] (pin 12), NSCS[2A] (pin 13), NSOE (pin 25), NSWE[1] (pin 26), NSWE[0] (pin 27), NSUB (pin 52),

- Set to Logic Low:

  SAMPCLK (pin 63), U2TXD (pin 76), U1TXD (pin 78),

In this mode, DISCIO can be configured to output a '1' to power down an RF Front-end IC (achieved if DISCIO_CFG[2:0] in IO_REV register is set to '100'. It is strongly recommended that if a RF Front-end power-down function is required, a 1kohm pull-down resistor should be connected to DISCIO. The reset condition of the GP4020 is to make DISCIO pin an input, which will mean that voltage on PDn input on RF Front-end could be >+0.8V if no resistor was present, and the RF IC could be disabled.

Note: the DISCIO configuration (i.e. output RF_PDOWN / input), is only reset by an NPOR_RESET event.

In the GP2010 and GP2015 GPS RF Front-end ICs, the Power-on Reset circuitry is NOT disabled by an RF Power Down, and so POWER_GOOD will still indicate a valid state, whilst main-power is still present. The POWER_GOOD signal originates from the RF Front-end, and the incidence of a POWER_GOOD failure will normally suggest that Main Power has been lost from the whole GPS receiver. On restoration of POWER_GOOD, there will be a delay of approx. 5ms while the RF Front-end PLL locks up to the 10.00MHz TCXO.

If POWG_EN is set to '0', the POWER_GOOD input will NOT power-down the GP4020.

12.6.2 RF Input and RF Front-end Power-Down

A Power-down of an RF Front-end IC, along with disabling the 40MHz Low Level Differential Input cell in the System Clock Generator, can be made to occur if:

a) RF_PD bit (POW_CNTL[0]) set to '1'.

b) RF_SLEEP bit (POW_CNTL[10]) set to '1'.

Under these conditions, the following pins on the GP4020 (100-pin package) get set as follows:

- Set as High Impedance (i.e. Tristate):

  NSCS[0] (pin 11), NSCS[1] (pin 12), NSCS[2A] (pin 13), NSOE (pin 25), NSWE[1] (pin 26), NSWE[0] (pin 27), NSUB (pin 52), SDATA[15:0]

- Set to Logic Low:

  SAMPCLK (pin 63), U2TXD (pin 76), U1TXD (pin 78)

In this mode, DISCIO (pin 55) can be configured to output a '1' to power down an RF Front-end IC (achieved if DISCIO_CFG[2:0] in IO_REV register is set to '100'.

As the 40MHz input from the RF Front End IC is disabled, this option should only be used if the Firefly clock source is not derived from this. Otherwise, the system will lock up, due to there being no clock-source for the Firefly MF1.

12.6.3 Real Time Clock Crystal Oscillator Cell Disable

The RTC 32kHz crystal-oscillator cell can be disabled by setting IDDQ_TEST (pin 70 (100-pin package)) to High. The Real Time Clock is NOT disabled by a Full Power Down.

(Note: Setting IDDQ_TEST to High is NOT a mode required in normal operation, and is primarily involved with manufacturing test of the GP4020. It is used to disable the Crystal Oscillator in the Real Time Clock, and the Processor Clock Crystal Oscillator, the PLL and the 40MHz Low Level Differential Input cell in the System Clock Generator).

12.6.4 System Clock Generator Processor Crystal Oscillator cell Disable

The Processor Crystal Oscillator cell in the System Clock Generator, can be disabled by setting PRX_EN bit (POW_CNTL[1]) to '0';

12.6.5 System Clock Generator Phase Locked Loop (PLL) Disable

The Phase-locked Loop cell in the System Clock Generator, can be disabled by:

1) Setting PLL_PD bit (POW_CNTL[2]) set to '1'.

2) Setting PLL_SLEEP bit (POW_CNTL[9]) set to '1'.

12.7 Peripheral Control Logic Registers

The Peripheral Control Logic uses four registers. Other blocks of circuitry in the GP4020 besides the PCL block use registers in the address space for the PCL. Reference to all registers in the PCL address space is shown in *Table 12.4 below*. Those Blocks actually used by the PCL are indicated with the Functional Block designation "PCL". All registers in *Table 12.4 below* can be accessed as either byte, half-word, or word.

The GP4020 Peripheral Control Logic Base Address is 0x4010 1000.

Address Offset	Register	Direction	Function	Function Block
0x000	*RTC_PRE*	*Read*	*Real Time Clock Pre-scaler value*	RTC
0x002	*RTC_SEC_B*	*Read*	*16 LSBs of Real Time Clock second counter*	RTC
0x004	*COMP_RTCP*	*Read /Write*	*Comparison value for Real Time Clock Pre-scaler*	RTC
0x006	*COMPS_RTCS*	*Read /Write*	*Comparison value for 8 LSBs of Real Time Clock Second counter, and 8 MSBs of Real Time Clock second counter*	RTC
0x008	POW_CNTL	Read /Write	Power Control Register	PCL
0x00A	PLL_CNTL	Read /Write	PLL Control Register	SCG
0x00C	IO_REV	Read /Write	Input / Output Multiplex configure	PCL
0x00E	IP_READ	Read	Read access to Input / Output signals	PCL
0x010	PER_STAT	Read /Write	Configure / monitor Interrupts and Resets	PCL / 1PPS
0x012	*TIC_RET*	*Read /Write*	*TIC Period slewing logic, and access a data retention register.*	1PPS
0x014	*TIM_DEL_LO*	*Read /Write*	*Timemark Delay Down-Counter (LSBs)*	1PPS
0x016	*TIM_DEL_HI*	*Read /Write*	*Timemark Delay Down-Counter (MSBs)*	1PPS

Table 12.4 Peripheral Control Logic Register Map

All registers are addressable as 32-bit locations only, but can use sub-memory access.

12.7.1 PCL Power Control Register - POW_CNTL - Memory Offset 0x008

This register is used to configure Power Control modes in the GP4020, and configure signal inputs and outputs for the PLL in the System Clock Generator. This is primarily a write-only register, but on reading the register, the settings made by the previous Write can be observed.

Bit No.	Mnemonic	Description	Reset Value	R/W
15	POWG_EN	Controls whether a power-fail indication due to POWER_GOOD input (Pin 64 (100-pin package)) should power-down the all GP4020 functions. '1' = Disable all functions except the Real Time Clock, and the Data Retention Register in the 1PPS Timemark Generator, when POWER_GOOD = '0' '0' = keep all functions enabled, irrespective of state of POWER_GOOD.	1	R/W
14	WATCH_EN	'1' = Enable Watchdog function at start-up. '0' = Disable	0	R/W
13	WAK_DISC	Controls whether a sleep function previously enabled can be disabled by setting a '1' on the DISCIP1 input (GPIO[4] pin 95 (100-pin package)), from an external source. '1' = Enable wake-up event due to DISCIP1 = High. '0' = Disable.	0	R/W
12	WAK_COR	Controls whether a sleep function previously enabled can be disabled by a correlator-sourced interrupt (ACCUM_INT or MEAS_INT). '1' = Enable wake-up event due to correlator interrupt. '0' = Disable.	0	R/W
11	WAK_UART	Controls whether a sleep function previously enabled can be disabled by data received on the input of UART2. '1' = Enable wake-up event due to UART2 received data. '0' = Disable.	0	R/W

Bit No.	Mnemonic	Description	Reset Value	R/W
10	RF_SLEEP	'1' = Disable 40MHz low-level differential input in System Clock Generator, and apply an active High power-down signal to the RF Front-end (via DISCIO (pin 55 (100-pin package)), if so configured (ref. IO_REV register). Can be re-enabled by a wake-up event. '0' = no effect	0	R/W
9	PLL_SLEEP	'1' = Disable and reset the PLL in System Clock Generator. Can be re-enabled by a wake-up event. (See Note) '0' = no effect	0	R/W
8	F_SLEEP	'1' = Disable the Firefly MF1 system clock (BμILD_CLK). Can be re-enabled by a wake-up event. '0' = no effect	0	R/W
7:6	B_CLK_SEL[1:0]	UART_CLK divider block selector. Allows selection of different output division ratios for the B_CLK signal, to allow small resolution changes in B_CLK frequency, if required. The divider ratio is set to divide by 1, in the reset condition. '00' = divide by 1 (i.e. through connection) '01' = divide by 2 '10' = divide by 4 '11' = divide by 8	00	R/W
5	PLL_BYP	PLL Bypass connection. Allows input signal to PLL to appear at input to B_CLK divider block, effectively removing PLL from signal path. The PLL is bypassed in the reset condition. '1' = Enable PLL bypass, remove PLL from the signal path. '0' = Disable PLL bypass, connect PLL.	1	R/W
4:3	PLL_IN_SEL[1:0] *(See Note)*	PLL Input (& PLL Bypass) signal selector. Allows either divided down versions of the M_CLK signal (20.0MHz or 10.0MHz) or a signal from the Processor Crystal Oscillator (10.0MHz to 16.0MHz) to be applied to the PLL CLKINB input as a PLL reference signal. M_CLK / 2 is applied to the PLL CLKINB input in the reset condition. '0x' = connect the output from the Processor Crystal Oscillator to PLL CLKINB input '10' = connect M_CLK / 2 (=20MHz) to PLL CLKINB input '11' = connect M_CLK / 4 (=10.0MHz) to PLL CLKINB input	10	R/W
2	PLL_PD	PLL Power Down. PLL is Disabled in the reset Condition '1' = disable the PLL immediately. '0' = Enable the PLL after a wait period of approx. 183μs (6 * 32kHz clock cycles, determined by the Real Time Clock block). This allows the CLKINB to stabilise.	1	R/W
1	PRX_EN	Enable Processor Crystal Oscillator block.. '1' = Enable the Processor Crystal Oscillator; start up time in 10ms typical '0' = disable the Processor Crystal Oscillator. Only to be done if B_CLK is derived directly from M_CLK.	1	R/W
0	RF_PD	Power Down RF Front-end and 40MHz Low Level Differential Block. Blocks are powered Up in the Reset Condition. '1' = disable the Differential Block and apply an active High power-down signal to the RF Front-end (via DISCIO (pin 55 (100-pin package)), if so configured (ref. IO_REV register). Should only be used if B_CLK is derived from the Processor Crystal Oscillator. '0' = re-enable the Differential Block and apply an active Low power-on signal to the RF Front-end (via DISCIO (pin 55 (100-pin package)), if so configured (ref. IO_REV register).	0	R/W

Table 12.5 PCL POW_CNTL Register

Note: For each change of value of PLL_IN_SEL[1:0] or at PLL wake-up, the PLL will be disabled for a wait period of approx. 183μs (6 * 32kHz clock cycles, determined by the Real Time Clock block). This allows the CLKINB to stabilise.

12.7.2 PCL Input / Output Control register - IO_REV - Memory Offset 0x00C

This register combines the control of the Input / Output signal multiplexing on the GPIO[7:0], MULTI_FNIO and DISCIO pins (Pins 91, 92, 93, 95, 96, 97, 99, 100, 54, 55 respectively in 100-pin package), and a read of the Chip Revision register. Bits 9:0 can be read as well as written to; by reading the bits, the settings made by the previous Write can be observed.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:10	CHIP_REV[5:0]	Read Only Chip Revision number (MSB = Bit 5). '000000' = Rev 0. First version of GP4020 '000001' = Rev 1. Second version , etc.	000000	R
9	EXT_NCS0	'1' = Disable Internal Boot ROM, at reset, if MULTI_FNIO (pin 54 (100-pin package)) is set High. '0' = Enable Internal Boot ROM, at reset, if MULTI_FNIO (pin 54 (100-pin package)) is set High.	0	R/W
8	DISCOP_MUX	'1' = Connect DISCOP output from 12-channel Correlator to GPIO[5] (pin 93).(*See **Note***) '0' = Connect GPIO[5] to GPIO[5] (pin 93).	0	R/W
7:6	BSIO_MUX[1:0]	Multiplex BSIO connections onto GPIO[3:0] pins. '00' = GPIO[3:0] connect to GPIO[3:0] input / output pins '01','10','11' = BSIOCLK connects to GPIO[0] (pin 100 (100-pin package)) BSIODATA connects to GPIO[1] (pin 99 (100-pin package)) 'x1' = BSIOSS[0] connects to GPIO[2] (pin 97) '1x' = BSIOSS[1] connects to GPIO[3] (pin 96)	00	R/W
5:3	MFNIO_CFG[2:0]	Multiplex signals onto MULTI_FNIO (pin 54). '0xx' = Input only '100' = 100kHz Square wave output (NOT Timemark aligned) '101' = UART_CLK output (i.e. Firefly system Clock, without sleep disable function) '110' = Low output (i.e. '0'); '111' = High output (i.e. '1')	000	R/W
2:0	DISCIO_CFG[2:0]	Multiplex signals onto DISCIO (pin 55). '0xx' = Input only '100' = RF_PD / RF_SLEEP output to Power-down pin on RF Front-end. '101' = TIC output '110' = Low output (i.e. '0'); '111' = High output (i.e. '1')	000	R/W

Table 12.6 PCL IO_REV Register

Note: DISCOP output can be used to output 100kHz Square wave Clock, Raw Timemark, High or Low outputs, as defined by the Correlator "SYSTEM_SETUP" register.

12.7.3 PCL Input Read register - IP_READ - Memory Offset 0x00E

This Read-only register allows the most recent state of a number of GP4020 input signals to be read.

Bit No.	Mnemonic	Description	Reset Value	R/W
15	PER_INT	PER_INT interrupt line, sourced by Peripheral Control Logic to Firefly MF1 core.	-	R
14	DISCIP1	DISCIP1 input to Correlator; can be accessed from GPIO[4] (pin 95 (100-pin package))	-	R
13	DISCOP	DISCOP output from Correlator; can be accessed on GPIO[5] using DISCOP_MUX (IO_REV register)	-	R
12	DISCIO	DISCIO input (pin 55 (100-pin package)), if DISCIO_CFG[2:0] in IO_REV register configured DISCIO as an input.	-	R
11	POWER_GOOD	POWER_GOOD input (Pin 64 (100-pin package))	-	R
10	MULTI_FNIO	MULTI_FNIO (pin 54 (100-pin package)), if MFNIO_CFG[2:0] in IO_REV register configured MULTI_FNIO as an input.	-	R
9	TIC	TIC output from 12-channel Correlator. Also available from bit 13 in Correlator register ACCUM_STATUS_B.	-	R
8	TIMEMARK	1PPS Timemark output (NOT Raw_Timemark)	-	R
7:6	*Reserved*		-	
5	MAG0	MAG0 data (pin 62 (100-pin package)) from RF Front-end.	-	R
4	SIGN0	SIGN0 data (pin 61 (100-pin package)) from RF Front-end	-	R
3	*Reserved*		-	
2	TESTMODE	Test mode input (pin 74 (100-pin package))	-	R
1	*Reserved*		-	
0	RF_PLL_LOCK	RF_PLL_LOCK input (pin 56 (100-pin package)) from RF Front-end.	-	R

Table 12.7 PCL IP_READ Register

12.7.4 PCL Status register - PER_STAT - Memory Offset 0x010

This register contains flags to indicate the source of a Reset signal, some control bits to disable some reset sources, some flags to indicate some Interrupt sources and some bits to disable them. Bits [15:11] & [7:5] can be read as well as written to; by reading the bits, the settings made by the previous Write can be observed.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:14	TIC_INT_EN[1:0]	Enable TIC_INT interrupt from 1PPS Timemark Generator. Control of the TIC_INT signal interrupt within the Overflow Control Block of the TIC Period slewing logic. '00' = Disable TIC_INT and RELOAD_TIC signals '01' = Enable TIC_INT and RELOAD_TIC signals. TIC_INT and RELOAD_TIC signals generated each time Modulo 7 adder overflows. Automatic TIC period extension occurs. ADJ_TIC bit (TIC_RET Register) set also. '10' = Enable TIC_INT and RELOAD_TIC signals. TIC_INT and ADJ_TIC bit (TIC_RET Register) set each time Modulo 7 adder overflows. A Read of ADJ_TIC and subsequent write to ADJ_TIC will set RELOAD_TIC and cause a TIC period extension. If ADJ_TIC not written to, no RELOAD_TIC generated and TIC will not be extended. '11' = Enable TIC_INT and RELOAD_TIC signals. TIC_INT set for every TIC, irrespective of adder overflow state. ADJ_TIC bit (TIC_RET Register) set each time Modulo 7 adder overflows. A Read of ADJ_TIC and subsequent write to ADJ_TIC will set RELOAD_TIC and cause a TIC period extension. If ADJ_TIC not written to, no RELOAD_TIC generated and TIC will not be extended.	00	R/W

Bit No.	Mnemonic	Description	Reset Value	R/W
13	POW_INT_EN	Enable PER_INT Interrupt signal to Interrupt Controller in Firefly MF1, due to POW_GD_INT (POWER_GOOD (pin 64) going Low.) '1' = Enable Interrupt due to POWER_GOOD Low '0' = Disable	0	R/W
12	RCMP_INT_EN	Enable PER_INT Interrupt signal to Interrupt Controller in Firefly MF1, due to RTC_CMP_INT signal from Real Time Clock. '1' = Enable Interrupt due to RTC_CMP_INT High. '0' = Disable	0	R/W
11	CLR_INT	Write: '1' = No effect. Write: '0' = Reset PER_STAT[10:8] after next UART_CLK cycle (i.e. Reset ALL Interrupt Read bits). Read: always '1'	1	R/W
10	TIC_INT (See Note1)	TIC_INT Interrupt status from 1PPS Timemark Generator. '1' = TIC period correction required / about to occur. (This bit will be set even if TIC_INT is disabled by PER_STAT[15:14]). '0' = No TIC period correction required / about to occur.	0	R/W
9	POW_GD_INT *(See Note1)*	POW_GD_INT Interrupt status from POWER_GOOD input (pin 64 (100-pin package)). '1' = PER_INT Interrupt due to POWER_GOOD Low. (This bit will be set only if POW_INT_EN (PERSTAT[13]) is Enabled.) '0' = No PER_INT interrupt due to POWER_GOOD.	0	R
8	RTC_CMP_INT (See Note1)	RTC_CMP_INT Interrupt status from Real Time Clock. '1' = PER_INT Interrupt due to RTC counters equal to RTC Comparison Registers. (This bit will be set only if RCMP_INT_EN (PER_STAT[12]) is Enabled. '0' = No PER_INT interrupt due to RTC.	0	R
7	EN_PLL_RST	Reset of Firefly and Correlator due to RF_PLL_LOCK (pin 56 (100-pin package)) going Low. '1' = Enabled '0' = Disabled ***Note:*** *This should only be disabled if UART_CLK and BμILD_CLK are NOT derived from M_CLK in the SCG block.*	1	R/W
6	EN_POW_RST	Reset of Firefly and Correlator due to POWER_GOOD (pin 64 (100-pin package)) going Low. '1' = Enabled '0' = Disabled	1	R/W
5	CLR_RST	Write: '1' = No effect. Write: '0' = Reset PER_STAT[4:0] after next UART_CLK cycle (i.e. Reset ALL "Reset-Source" Read bits). Read: always '1'	1	R/W
4	SFT_RESET *(See Note 2)*	Software triggered Reset of Hardware. Write '1' = No effect. Write '0' = Reset of Firefly and Correlator at next UART_CLK cycle. Read '1' = Reset due to SFT_RESET (writing '0' to this bit), has occurred since last CLR_INT or NSRESET clear-event.	Rd: 0 Wr: 1	R/W
3	PLL_RESET *(See Note 2)*	'1' = Reset due to RF_PLL_LOCK = Low, has occurred since last CLR_RST or NSRESET clear-event. '0' = No reset event due to RF_PLL_LOCK has occurred	0	R

Bit No.	Mnemonic	Description	Reset Value	R/W
2	POW_RESET *(See Note 2)*	'1' = Reset due to POWER_GOOD = Low, has occurred since last CLR_RST or NSRESET clear-event. '0' = No reset event due to POWER_GOOD has occurred	0	R
1	WAT_RESET *(See Note 2)*	'1' = Reset due to Watchdog Time-out, has occurred since last CLR_RST or NSRESET clear-event. '0' = No reset event due to Watchdog has occurred	0	R
0	NSRST_RESET *(See Note 2)*	'1' = Reset due to NSRESET (pin 75 (100-pin package)) = Low, has occurred since last CLR_RST clear-event. '0' = No reset event due to NSRESET has occurred	0	R

Table 12.8 PCL PER_STAT Register

Notes:

1) When PER_INT received from PCL, the software Interrupt Service Routine should read bits 10:8 to determine which interrupt source has caused the interrupt to occur.

2) Bits 4:0 reset by setting CLR_RST ='0' or NSRESET='0' only. All other reset sources have no effect.

13 REAL TIME CLOCK (RTC)

13.1 Introduction

The Real Time Clock (RTC) is used to provide an incremental time indicator. It is based on a 32768Hz watch-crystal oscillator, with a 15-bit divider to produce a 1Hz clock, and a 24-bit counter accumulator to count seconds from zero upto a maximum of 194days (approx. 6 months). The RTC operates on the principle of incremental elapsed seconds, rather than absolute time. Consequently, the RTC does NOT store dates on the Gregorian Calendar, and is hence immune from the Year 2000 rollover issue.

Figure 13.1 below shows a block diagram of the Real Time Clock.

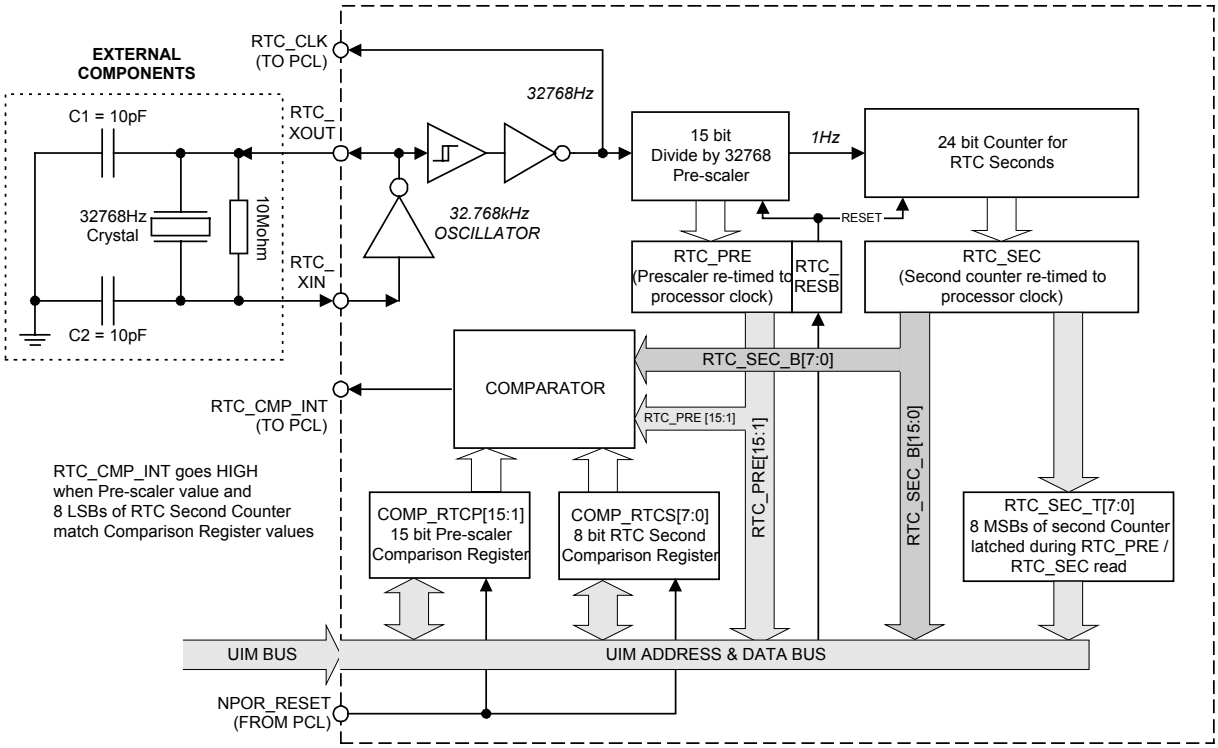


Figure 13.1 Real Time Clock Block Diagram

The RTC incorporates some comparison registers (with a comparator), which allow an interrupt signal to be generated when a pre-programmed time is reached by the RTC. The comparison registers use the full 15-bits from the RTC pre-scaler and the eight least-significant bits from the RTC second counter, to give a range of effectively 23-bits with a 32.768kHz clock. This will give a wide-range programmable delay of between 30.5μs and 256s. The RTC_CMP_INT signal goes High when the stored comparison value equals the value accumulated by the RTC. The RTC_CMP_INT signal is cleared by writing a '0' to CLR_INT (PER_STAT[11] in the PCL block).

13.2 32kHz Crystal Oscillator

The 32kHz Crystal Oscillator will operate with a standard watch crystal (32768Hz = 2^{15}Hz). It is based on a Pierce Oscillator Cell, (Refer to *Section 14.3 "Processor Crystal Oscillator" on page 137*, for more information). The oscillator output is via a buffered Schmitt trigger, which provides a square wave, which is compatible with the dividers, counters and registers in the Real Time Clock cell.

The Pierce oscillator relies on a resistor to provide negative feedback, which is set by a resistor. The 32kHz oscillator requires a resistor of high value (10MΩ), and is needed externally to the GP4020. A 32kHz crystal has a high Effective Series Resistance (ESR), in the region of 40kΩ. The nominal loading capacitors C1 and C2 for the

crystal need to be set to ensure that the total loop gain of the oscillator is high enough to guarantee continuous oscillation under all conditions. Normally this will mean that a loop gain of greater than 1 is needed.

A set of equations for calculating C1 and C2 are explained in Section *14.3 "Processor Crystal Oscillator" on page 137*.

In the case of this 32.768kHz oscillator, key parameters are:-

Transconductance (gm) = 9.56uA/V

Output impedance (Zout) = 422MΩ

Feedback Resistor (Rf) = 10MΩ

Using the equations highlighted above, the nominal value for the crystal loading capacitors, C1 and C2 should be between 5.6pF and 10pF.

13.3 Real Time Clock Registers

The Real Time Clock uses four registers. These registers are addressable in the same part of the memory map as the Peripheral Control Logic Block (PCL).

The GP4020 Real Time Clock Base Address is 0x4010 1000.

All registers are addressable as 32-bit locations only, but can use sub-memory access.

Address Offset	Register	Direction	Function
0x000	RTC_PRE	Read/Write	Read Pre-scaler divider value
0x002	RTC_SEC_B	Read	Read Least significant 16-bits of Real Time Clock 24-bit second counter
0x004	COMP_RTCP	Read/Write	Comparison value for Real Time Clock Pre-scaler
0x006	COMPS_RTCS	Read/Write	Comparison value for 8-bits of Real Time Clock second counter & Read only access to Most significant 8-bits of Real Time Clock 24-bit second counter

Table 13.1 Real Time Clock Register Map

Whilst most of the registers in the GP4020 can be reset by a reset event (POWER_GOOD, RF_PLL_LOCK, SFT_RESET, NSRESET and Watchdog), there are a couple of areas where a reset will only occur by writing a '0' to bit 0 of the RTC_PRE register:

1) RTC_PRE[15:1] in the RTC_PRE register;
2) All data bits in register RTC_SEC_B;
3) RTC_SEC_T[7:0] in the COMPS_RTCS register.

13.3.1 RTC Pre-Scaler Divider Register - RTC_PRE - Memory Offset 0x000

A read of this register returns the number of accumulated RTC clock cycles in the 15-bit RTC pre-scaler, at the time of access, upto a value of 2^{15} RTC clock cycles (the pre-scaler will rollover once every second). At the time this register is accessed, the values in register RTC_SEC_T (part of Register COMPS_RTCS) will be latched.

NOTE: Only a write of '0' to Bit 0 of register RTC_PRE will reset the 15-bit RTC Pre-scaler divider (RTC_PRE[14:0]) and 24-bit RTC counter (RTC_SEC_B and RTC_SEC_T[7:0]). No other form of reset will clear these values.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:1	RTC_PRE[15:1]	Number of RTC clock cycles at sample time, within 1 second since last divider reset/rollover. (One clock cycle = 1/32768 = 30.5μs). Most Significant Bit = Bit 15. **Note: This data ONLY reset by writing '0' to bit 0 of this register. NOT resettable by any other reset source.**	0x0000	R
0	RTC_RESB	Write '0': Reset RTC Counter and divider Write '1': No effect Read: Always read '0'	0	R/W

Table 13.2 RTC_PRE Register

13.3.2 RTC Accumulated Seconds - 16 LSBs - RTC_SEC_B - Memory Offset 0x002

Readable only. A read of this register returns the least significant 16-bits of the accumulated RTC seconds in the 24-bit second counter, at the time of access, upto a value of 2^{24} seconds = 16.7Mseconds = 194days. The counter will stop counting when all 24-bits are set to '1'. At the time this register is accessed, the values in register RTC_SEC_T (part of Register COMPS_RTCS) will be latched.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	RTC_SEC_B	16 LSBs of accumulated RTC seconds since last 24-bit counter reset. Most Significant Bit = Bit 15 **Note: This data ONLY reset by writing '0' to bit 0 of RTC_PRE. NOT resettable by any other reset source.**	0x0000	R

Table 13.3 RTC_SEC_B Register

13.3.3 RTC Pre-Scaler Comparison Register - COMP_RTCP - Memory Offset 0x004

This is a Read / Write register used to set a Comparison value (15-bits) which is compared with the accumulated value in the 15-bit RTC Pre-scaler by means of a comparator block. When the value in the comparison register and the RTC Pre-scaler are equal, and the values in the COMP_RTCS register equals the 8 LSBs of the RTC Counter, an Interrupt signal RTC_CMP_INT is produced. The comparator is disabled during a write to this register.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:1	COMP_RTCP[15:1]	15-bit RTC Pre-scaler Comparison Value. Most Significant Bit = Bit 15.	0x7FFF	R/W
0	COMP_RTCP[0]	Read: Always read '0'	0	R

Table 13.4 RTC COMP_RTCP Register

13.3.4 RTC Second Comparison Register - COMP_RTCS - Memory Offset 0x006

This is a dual-purpose register.

The Read Register [15:8] holds RTC_SEC_T[7:0]; the latched value of the 8 MSBs of the accumulated RTC seconds in the 24-bit second counter, at the time of access, upto a value of 2^{24} seconds = 16.7Mseconds = 194days. The counter will stop counting when all 24-bits are set to '1'.

The Read / Write register [7:0] is used to set a Comparison value (8-bits) which is compared with the accumulated value in the 8 LSBs of the RTC Second Counter by means of a comparator block. When the value in the comparison register and the RTC Second Counter are equal, and the values in the COMP_RTCP register equals the 15-bits of the RTC Pre-scaler, an Interrupt signal RTC_CMP_INT is produced. The comparator is disabled during a write to this register.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:8	RTC_SEC_T[7:0]	8 MSBs of accumulated RTC seconds since last 24-bit counter reset. Most Significant Bit = Bit 15 **Note: This data ONLY reset by writing '0' to bit 0 of RTC_PRE. NOT resettable by any other reset source.**	0x00	R
7:0	COMP_RTCS[7:0]	8-bit RTC Counter Comparison Value. Most Significant Bit = Bit 7.	0xFF	R/W

Table 13.5 RTC COMP_RTCS Register

14 SYSTEM CLOCK GENERATOR (SCG)

14.1 Introduction

The System Clock Generator (SCG) is used to generate two clock signals for the GP4020:

- The UART_CLK which runs UART2 continuously and produces the BμILD Clock. The BμILD Clock runs all the BμILD bus components, including the Firefly MF1 core with the ARM7TDMI microprocessor via a disable gate in the Peripheral Control Logic Block.
- The Correlator Master Clock (M_CLK).

The UART_CLK is used to generate the microprocessor system clock. It can be derived from two sources (RF Front end or a Processor Crystal Oscillator) and the frequency can be programmed to suit the overall system requirements, with the use of an on-chip PLL.

The M_CLK is essentially a 40MHz clock, which is phase-locked to the RF Front-end. The CLK_T and CLK_I signals from the RF Front-end are used to provide M_CLK in conjunction with a differential input amplifier. The M_CLK cannot be derived from any other source.

Figure 14.1 below shows a block diagram of the System Clock Generator.

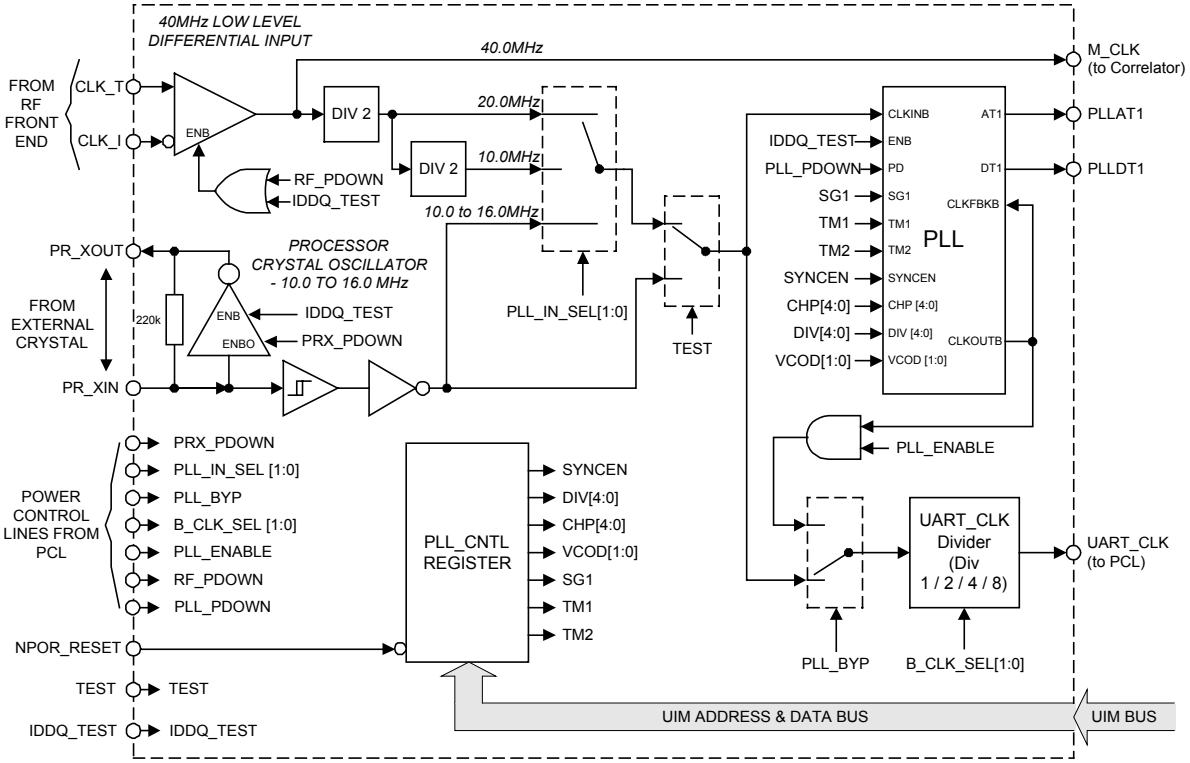


Figure 14.1 System Clock Generator Block Diagram

The System Clock Generator features two input sources:

- 40MHz low-level differential clock signal from an RF Front-end
- Processor Crystal Oscillator, for crystal frequencies between 10.0MHz and 16.0MHz.

14.2 40MHz Low Level Differential Input

The 40MHz low-level differential input can process the 40MHz signal from a RF Front-end chip. The signal should have a DC bias of less than approx. 1.7V with respect to GND, and the 40MHz signal should have amplitude of approx. 100mV at a frequency of 40MHz. As a minimum, the CLK_I and CLK_T signals from the RF Front-end should be present in order to provide a phase-locked 40MHz M_CLK clock signal to the 12-channel correlator block. This same M_CLK signal can also be used to generate the UART_CLK (and BμILD_CLK) signal for all the other GP4020 system blocks, in conjunction with a series of clock selection logic and a Phase-locked-Loop (PLL).

If using the GP4020 in conjunction with a GP201x RF Front-end, care should be observed to ensure that the DC bias applied to the CLK_T (pin 58 (100-pin package)) and CLK_I (pin 59) inputs of the GP4020 is set correctly. The maximum output bias of the GP201x RF Front end on the OPCLK+ and OPCLK- signals is Vcc - 0.8V, which could give a maximum bias of +2.8V (Vccmax = +3.6V).

It is recommended that the circuit shown in *Figure 14.2 below* is used to interface the OPCLK+ / - lines of the GP2015 RF Front-end to the CLK_T and CLK_I inputs on the GP4020. The PSU configuration for the GP4020 allows the GP4020 to remain powered (using "GP4020 +3.3V") while the remaining circuitry within the GPS Receiver is powered off (using "Main +3.3V"). The use of 100kΩ resistors is to ensure that the standby current through the resistors is moderately low (~20μA) while the receiver is powered up, but when "Main +3.3V" is removed, the current through these resistors will disappear. Also, the A1Vdd pin (Pin 57 (100-pin package)) and the Vdd supply to the 100kΩ resistors needs to be well de-coupled via wide-band de-coupling to the GND pin at pin 60. This is to ensure that supply noise is minimised, and that the 40MHz low-level differential input block does not oscillate. Note that the 100kΩ resistors should be AC-locked to the A1Vdd pin, by means of a 10nF coupling capacitor between "GP4020 +3.3V" and "Main +3.3V", in close proximity to the A1Vdd pin itself.

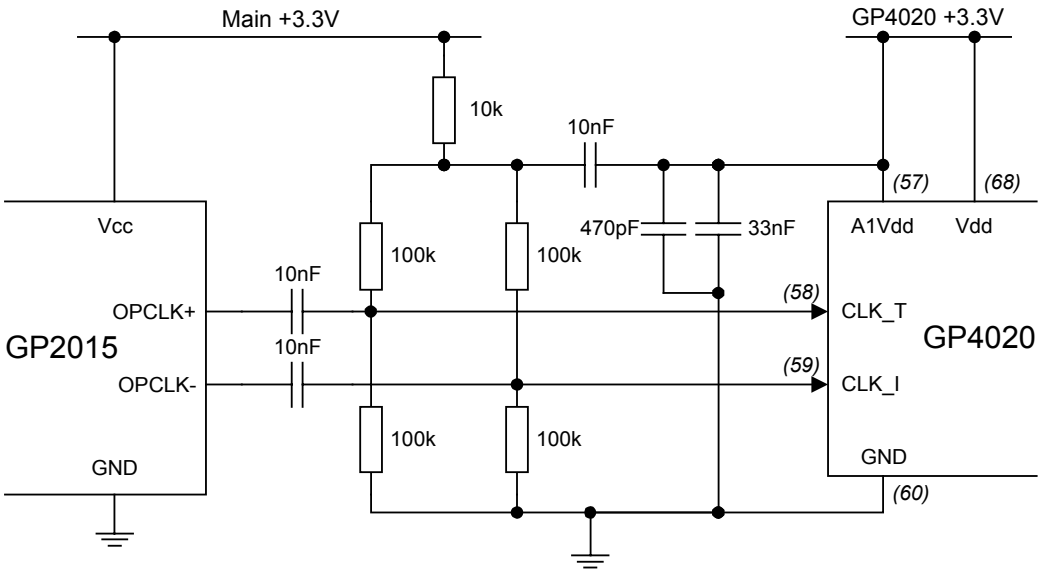


Figure 14.2 Circuit to interface OPCLK+/- from GP2015 to CLK_T / _I on GP4020

The 40MHz low-level differential block, which generates M_CLK to the correlator core (and optionally the Firefly MF1 core), is disabled during RF sleep mode, or by RF_PD being high. The power down signal to the differential cell can also be output on DISCIO pin (if DISCIO configured to output RF_PDOWN, it will be high during power down). If the differential cell is powered down due to RF_SLEEP, it will be enabled again by a wake-up event. If the differential cell is powered down by RF_PD bit in the POW_CNTL register, the processor has to manually enable it. Therefore, if BμILD_CLK to the Firefly MF1 core is derived from M_CLK, the RF_SLEEP mode should not be used.

14.3 Processor Crystal Oscillator

The Processor Crystal Oscillator may need to be used with an external crystal, to generate the clock source for the UART_CLK signal. The two instances where this may be necessary are:

1) If the RF Front-end is to be used in a Power-down or sleep mode; powering down the RF Front-end will disable the 40MHz from the RF Front-end and also disable the 40MHz Low level Differential block, leaving the GP4020 potentially locked up without any system clock;

2) To produce a system clock at a strategic frequency that cannot be derived from the PLL using the input clock reference from the M_CLK source (10.0MHz or 20.0MHz).

The Processor Crystal Oscillator is switched in by default when TEST (pin 67 (100-pin package)) is set High. This is for test purposes only.

The Processor Crystal Oscillator will operate with Crystals over the frequency range of 10.0MHz to 16.0MHz. It is based on a Pierce Oscillator Cell, as shown in *Figure 14.3 below*. Output from the cell is via a buffered Schmitt trigger, which provides a square wave, which is compatible with the internal clock requirements of the GP4020, including the PLL block.

Once a crystal has been chosen in the range 10.0 to 16.0MHz, the load capacitors C1 and C2 can be selected. The capacitor values ensure correct operation of the pierce oscillator such that the total loop gain is greater than unity. Correct selection of the two capacitors is very important and the following method is recommended to obtain values for C1 and C2.

Although oscillation may still occur if the loop gain is just above 1, a loop gain of 5 is recommended. This is to ensure that oscillations will occur across all variations of temperature, voltage supply and silicon process batch. In addition, this ensures that the circuit will exhibit a reliable start-up performance.

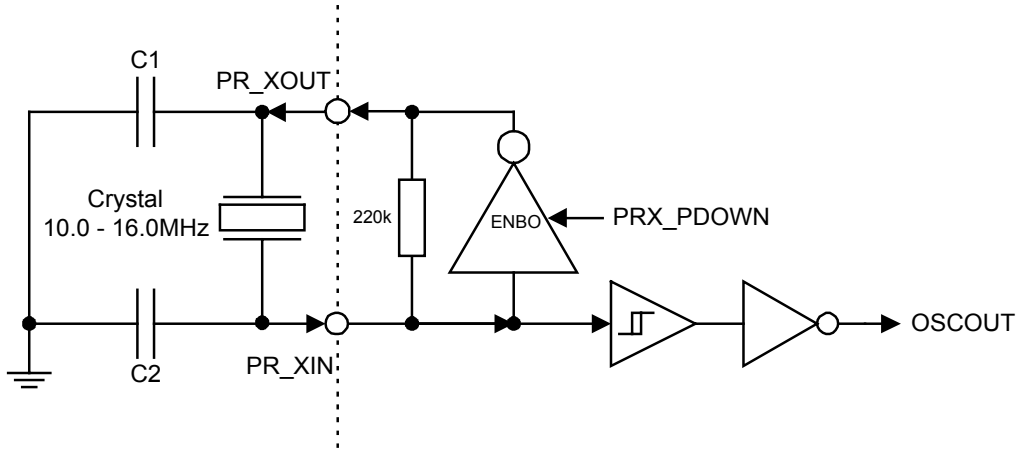


Figure 14.3 Processor Clock Oscillator, crystal connection configuration

$$A = \frac{C_{out} g_m}{C_{in}} \left[\frac{(C_{out} + C_{in})}{R_f C_{in}} + \frac{1}{Z_{in}} + \frac{1}{Z_{out}} \right]^{-1} \quad \text{..... 1}$$

$$Z_{in} = \frac{1}{(2\pi f C_{out})^2 \times ESR} \quad \text{..... 2}$$

where: A = Total Loop Gain (goal = 5)

C_{in} = Capacitance on PRX_IN pin = (C1 + 10pF)

C_{out} = Capacitance on PRX_OUT pin = (C2 + 10pF)

Z_o = Output Impedance of oscillator at PRX_OUT

G_m = Transconductance of oscillator

R_f = Feedback Resistor (on-chip)

ESR = Equivalent Series Resistance of crystal

F = Fundamental Crystal frequency

Equations 1 and 2 above can be used to calculate the range of tolerable crystal capacitance values, when the crystal characteristics are known (frequency and ESR). Typical values for C1 and C2 for a 10.0MHz crystal will be 47pF each, and for a 16.0MHz crystal will be 39pF each. This assumes a crystal ESR of approx. 25Ω, output impedance of 110kohms and an oscillator transconductance of 2.24mA/V.

The capacitor values used with the Processor Crystal Oscillator are NOT critical, and calculations of values will not normally be required. However, some crystals may have some exceptional ESR values, or more appropriately, the range of oscillator coefficients over all environmental conditions may be a concern.

14.3.1 Using the Processor Clock Oscillator with an external frequency source

Some GPS receiver systems may use an external oscillator (TCXO) to generate a PLL reference for the RF Front-end device. There may be instances where the Processor Clock Oscillator may need to be used when the RF Front-end is powered down, and the cost of an additional crystal for the GP4020 is deemed an unacceptable extra expense.

The PRX_IN (pin 66 (100-pin package)) input to the processor Clock oscillator is NOT a true CMOS input, due in part to the on-chip feedback resistor that is used with the oscillator itself. Because of this, the PRX_IN input has a high input capacitance (approx. 20pF), which will mean that the input impedance will be low at a frequency of 10MHz (approx. 800ohms). Hence, the drive from the TCXO will need to be low-impedance to ensure that adequate signal triggers the Processor Clock buffer.

The GP4020 presents a noisy load to any analogue signals connected to it. As the TCXO drives the RF Front-end PLL, it is imperative to ensure that the RF PLL does not get interference via the TCXO from the GP4020. The GPS system performance can be severely degraded if the RF Front-end cannot produce an accurate set of Local-Oscillator signals for the IF down-conversions. Therefore, it is important to ensure that the quality of the TCXO is NOT degraded by the GP4020.

Consequently, it is recommended that if the TCXO is going to be used to provide a backup system clock while an RF Front-end is powered down, that some active buffering be introduced between the TCXO and the GP4020 PRX_IN input. *Figure 14.4 below* shows the generic connection scheme of a TCXO to both a GPS RF Front-end (e.g. GP2015), the GP4020, and a recommended TCXO buffer circuit, using a CMOS inverter gate. The gate uses a feedback resistor in order to bias the gate input to be approx. mid-rail (1.65V DC) when no external signal is driving it. Note that the TCXO uses a supply resistor and some large capacitors to assist with Power Supply Rejection, particularly from digital interference on the power-supply at low frequencies.

The TCXO buffer circuit is a biased high-speed CMOS logic gate. As the gain of a Logic Gate is VERY high, the output from it is essentially a square-wave, which gives a very good drive signal for the GP4020. This results in a low-jitter BμILD_CLK signal. It is very important to ensure that the Logic Gate is powered from the same PSU line as the TCXO, in order to offer the highest degree of buffering between TCXO and GP4020.It may be appropriate to use a single-gate Logic chip for the TCXO buffer. Many logic gates are now available not only in quantities of 4 or 6 in a standard 14-pin package, but also as single gates in SOT-23 style packages.

Care should be used to minimise the effects of radiated 10MHz harmonics when using a TCXO buffer. Many buffers have bandwidths well into 100s of MHz. Poor layout of tracks from this buffer in a GPS Receiver could produce radiated interference at harmonics of 10MHz. This could affect the GPS RF Front-end sensitivity, and create EMC problems. It may be necessary to introduce a series resistor (approx. 33Ω or more) between the TCXO buffer output and the PRX_IN pin on the GP4020, to reduce the harmonic energy from the TCXO Buffer.

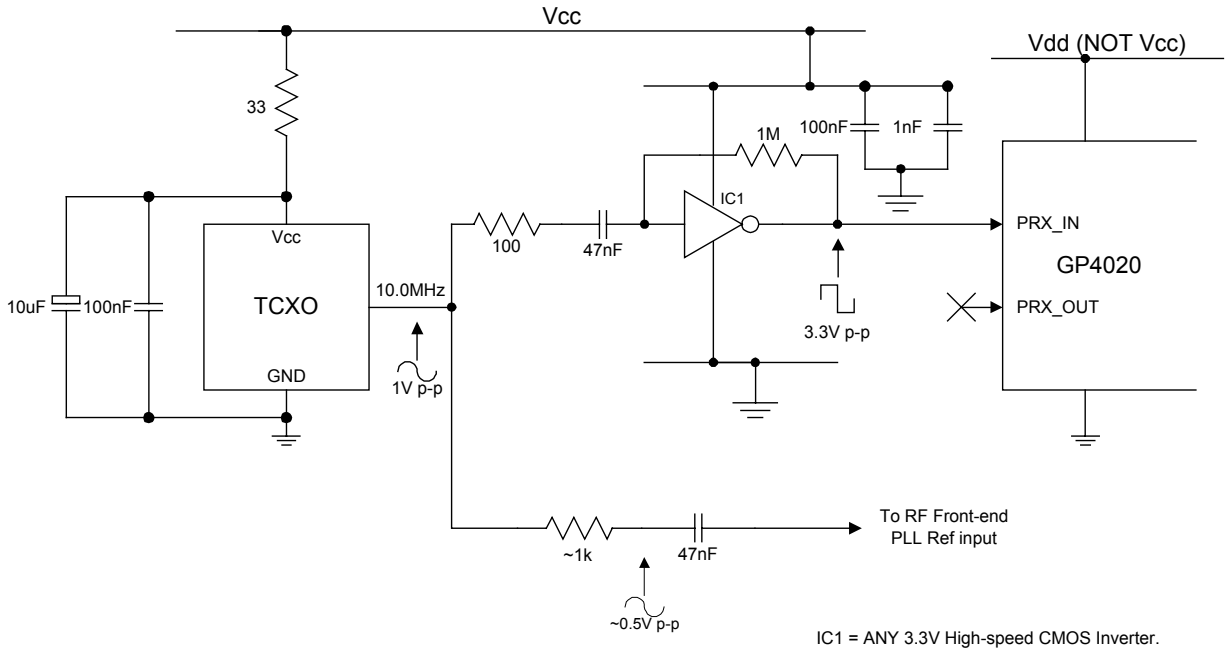


Figure 14.4 Connections of a TCXO frequency reference to the GP4020 Processor Crystal Oscillator

14.4 Phase Locked Loop (PLL)

14.4.1 Features

- Output clock frequencies from 10MHz to 250MHz
- Phase alignment offset: 0.3ns
- Phase alignment jitter: 0.5ns
- Low power consumption: 7mW at 20MHz input, 80MHz output frequency
- Internal programmable divider for clock multiplication between 1 and 25
- Integrated loop filter

14.4.2 PLL Principles and Operation

At the heart of the PLL is a phase comparator, charge pump, filter and VCO. These blocks are connected together, as shown in *Figure 14.5 below*, to produce a PLL system:

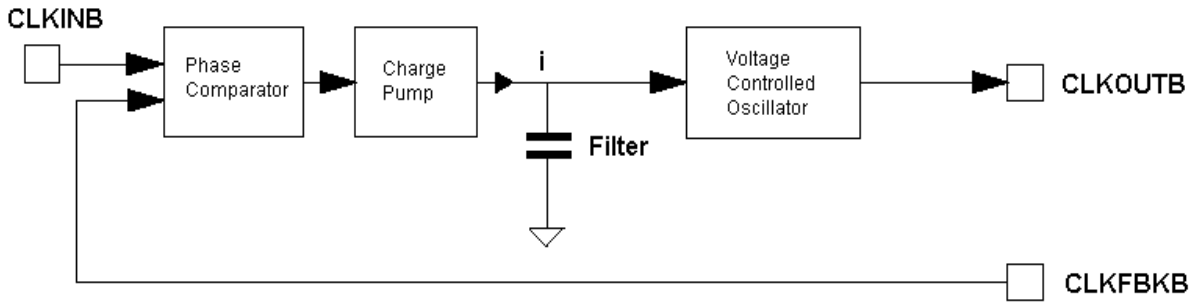


Figure 14.5 GP4020 System Clock Generator PLL Configuration

The PLL can provide accurate phase alignment between a generated clock and a reference clock without incurring delays normally associated with buffering. The PLL will phase lock the output clock 'CLKOUTB' to the reference-input clock 'CLKINB'. If the Voltage Controlled Oscillator (VCO) is set for the correct operating frequency range, and the loop has been made stable by the correct choice of the charge pump current, then the loop will accurately adjust the VCO to align the falling edges of the 'CLKINB' and 'CLKFBKB' inputs using the phase comparator. This phase alignment will be subject to a small average offset and a small amount of jitter due to noise sources.

The PLL in the GP4020 has been configured for Clock Multiplication, where the output frequency is a multiplied version of the reference input clock frequency. To achieve clock multiplication, a programmable divider is used in the feedback path of the PLL, as shown in *Figure 14.6 below*. A 5-bit programmable divider has been designed into the macro-cell. By using the clock multiplication mode of the PLL it is possible to generate a whole range of output frequencies of integer multiples of the input clock.

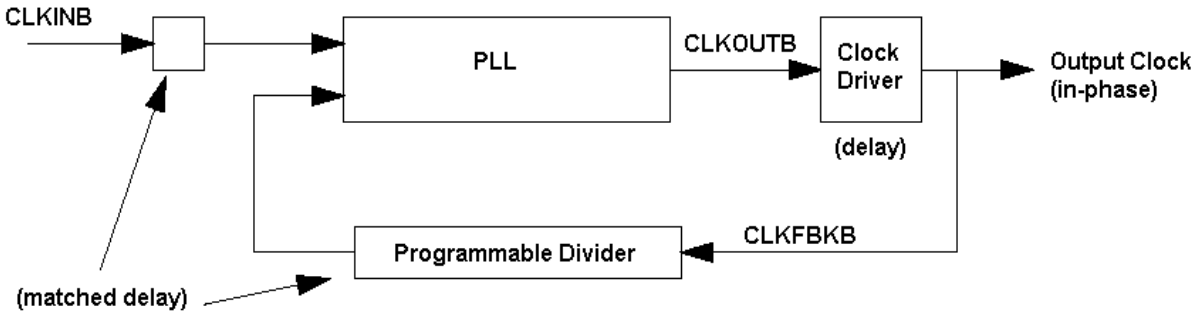


Figure 14.6 PLL Programmable Divider Configuration

Either the output from the Processor Crystal Oscillator or a divided version of M_CLK can be used as the input to a Phase Locked Loop (PLL). The output of the PLL can then be used to generate UART_CLK and subsequently BμILD_CLK for the Firefly MF1 Core.

The PLL can be disabled by PLL_PD being high, or by PLL sleep mode. The clock out of the PLL will remain disabled for 6 * 32kHz clock cycles (from the Real Time Clock Block) after PLL enabled (to ensure the PLL is stable before it is used to clock 'Firefly'). The PLL output will also be disabled for 6 * 32kHz clock cycles by any reset source, or due to PLL_IN_SEL changing.

PIN	DESCRIPTION
CLKINB	PLL input clock reference pin.
CLKFBKB	Clock feedback pin. This pin is connected to a node on the chip (usually the output of a clock buffer) via an inverting gate that is required to be phase aligned with the input clock. All phase synchronisation is to the falling edges of 'CLKFBKB' (these are the rising edges of the inverter input). The signal on 'CLKFBKB' is always be derived from 'CLKOUTB' when the PLL is in normal operation. If the feedback link is broken then the PLL will lock up at a high frequency.
CLKOUTB	PLL clock output pin. This pin is used to drive the device's clock buffer after inversion.

PIN	DESCRIPTION
VCOD[1:0]	PLL VCO Output Frequency Range selection pin. This input determines which of the 4 frequency ranges are selected. '00' operates the VCO between 80MHz and 250MHz; '01' operates the VCO between 40MHz and 125MHz; '10' operates the VCO between 20MHz and 63MHz; '11' operates the VCO between 10MHz and 32MHz
PD	PLL power down and reset pin. When PD is high the PLL is powered down, the 'CLKOUTB' output is forced high and all cells within the PLL are reset.
ENB	PLL Enable Bar pin. This pin will power down the PLL if taken high identically to the PD pin.
DIV [4:0]	Programmable divider programming bits used for setting up the PLL in clock multiplication mode. These inputs are binary weighted to give divider settings from 2 to 25. The binary value of 'n' will give a divider setting of N = n+2. For clock synchronisation (divide by 1) set SYNCEN to a '1' and DIV0-4 all to a '1' (to minimise power consumption).
SYNCEN	The PLL is forced into a clock synchronisation mode when pin SYNCEN is tied 'high' overriding the programming bits DIV0-4.
CHP [4:0]	Charge pump current setting pins. The internal charge pump current is set by the digital value on these pins (binary weighted; LSB=CHP0). The charge pump must be correctly programmed to ensure stability of the PLL control loop. Refer to *Table 14.2 on page 142, Table 14.3 on page 143 and Table 14.4 on page 144* for recommended values.
SG1	PLL test control pin.
TM1	PLL test pin 1.
TM2	PLL test pin 2.
PLLAT1	Analog Test Access Pin. During normal operating modes this signal is pulled low by the PLL. It should not be connected externally to the device.
PLLDT1	Digital Test Access Output pin. The state of this pin must be observable from the device pins during testing of the PLL.
PLLGND	PLL GND pin. This pin connects to the PLL GND supply.
PLLVDD	PLL VDD pin. This pin connects to the PLL VDD supply.

Table 14.1 PLL Block Pin Names & Descriptions

14.4.3 PLL Programming

The PLL charge pump (CHP[4:0]) and feedback divider (DIV[4:0] & SYNCEN) values must be programmed correctly according to the VCO range selected (VCOD[1:0]) and the clock multiplication ratio.

The maximum output frequency from the PLL can be up to 250MHz. The PLL output frequency will be an integer multiple of the PLL reference input frequency. The valid frequency input to the PLL from M_CLK can be 10.0MHz, 20MHz or any frequency from 10.0MHz to 16.0MHz as determined by an external crystal connected between PR_XIN and PR_XOUT on the Processor Crystal Oscillator.

Although the maximum system frequency is going to be much less than the maximum frequency of operation of the PLL block (250MHz), the wide range of the PLL frequency output gives a wide range of possible UART_CLK output frequencies, in conjunction with a programmable output stage divider. The smallest step resolution that the SCG has, is 1.25MHz by virtue of the output divider set to divide by 8, and that the lowest input reference frequency that the PLL can operate at is 10.0MHz.

The range of clock frequencies, which can be provided for UART_CLK, using a reference signal derived from M_CLK, is quite extensive. As an example, the frequencies that can be produced for UART_CLK when the M_CLK signal derived from the CLK_I and CLK_T signals (40MHz from a RF Front-end) are shown in *Table 14.2 below*.

Additional frequencies beyond the specified maximum speed for the GP4020 are also shown in *Table 14.3 on page 143*. Whilst in normal operation, it is not recommended to exceed the maximum specified operating frequency, the numbers in this table are shown for experimental purposes. Whilst under certain conditions it may be possible to run GP4020 above 30MHz, it is not recommended.

Note: Refer to the GP4020 Datasheet (DS5134) for specified limits on the GP4020 maximum operating speed.

If you intend to change the frequency of the PLL on the fly during time-critical code-execution, care should be used to ensure that the PLL is allowed to stabilise before allowing the code execution to continue. It is recommended that the software waits for the period specified by the "Worst-case settling time" parameter in *Table 14.2 on page 142, Table 14.3 on page 143 and Table 14.4 on page 144* after the new frequency has been programmed in. This will allow the PLL to lock to the new frequency required and reduce time-related jitter to a minimum.

Although the existence of a PLL in the System Clock Generator allows a flexible range of UART_CLK and BμILD_CLK frequencies to be produced, it should be pointed out that there will be harmonic spurious produced for any frequency selected. This can potentially produce spurious radiation at any of the RF Front-end IF frequencies, or, in some cases in the Receive RF L1 band. Whilst every attempt has been made to ensure that the GP4020 will respond at many key UART_CLK frequencies, it cannot be ruled out that there will be values where self-generated interference occurs.

Therefore, care should be used to ensure that any value of UART_CLK frequency selected, in conjunction with the wait-state properties of any external memory components, does not produce in-band radiation at any of the following key GP2015 RF Front-end frequencies:

1) 1575.42MHz ± 2.0MHz

2) 175.42MHz ± 2.0MHz

3) 35.42MHz ± 2.0MHz

UART_CLK O/P Freq. (MHz)	I/P Freq. MHz	PLL Mult Fact	Prog. Divider setting DIV [4:0]	Charge Pump setting CHP [4:0]	PLL SYNC MODE SYN CEN	PLL O/P VCO Freq. MHz	VCO Freq. Range VCOD [1:0]	BY-PASS PLL PLL_BYP	PLL O/P Divide Factor	B_CLK_SEL [1:0]	Ts[3] (μs)
1.25	10.0[1]	N/A	N/A	N/A	N/A	N/A	N/A	1	8	11	N/A
2.5	10.0[1]	N/A	N/A	N/A	N/A	N/A	N/A	1	4	10	N/A
3.75	10.0[1]	3	00001	01000	0	30	10	0	8	11	47
5.0	10.0[1]	N/A	N/A	N/A	N/A	N/A	N/A	1	2	01	N/A
6.25	10.0[1]	5	00011	00111	0	50	01	0	8	11	52
7.5	10.0[1]	3	00001	01000	0	30	10	0	4	10	47
8.75	10.0[1]	7	00101	01001	0	70	01	0	8	11	44
10.0	10.0[1]	N/A	N/A	N/A	N/A	N/A	N/A	1	1	00	N/A
11.25	10.0[1]	9	00111	00110	0	90	00	0	8	11	58
12.5	10.0[1]	5	00011	00111	0	50	01	0	4	10	52
13.75	10.0[1]	11	01001	00111	0	110	00	0	8	11	52
15.0	10.0[1]	3	00001	01000	0	30	10	0	2	01	47
16.25	10.0[1]	13	01011	01001	0	130	00	0	8	11	44
17.5	10.0[1]	7	00101	01001	0	70	01	0	4	10	44
18.75	10.0[1]	15	01101	01010	0	150	00	0	8	11	41
20.0	20.0[2]	N/A	N/A	N/A	N/A	N/A	N/A	1	1	00	N/A
21.25	10.0[1]	17	01111	01011	0	170	00	0	8	11	38
22.5	10.0[1]	9	00111	00110	0	90	00	0	4	10	58
23.75	10.0[1]	19	10001	01101	0	190	00	0	8	11	35
25.0	10.0[1]	5	00011	00111	0	50	01	0	2	01	52
26.25	10.0[1]	21	10011	01110	0	210	00	0	8	11	33
27.5	10.0[1]	11	01001	00111	0	110	00	0	4	10	52
28.75	10.0[1]	23	10101	01111	0	230	00	0	8	11	32
30.0[4]	10.0[1]	6	00100	01000	0	60	01	0	2	01	47

Table 14.2 Valid UART_CLK frequencies that can be produced from M_CLK (from RF Front end)

UART_CLK O/P Freq. (MHz)	I/P Freq. MHz	PLL Mult Fact	Prog. Divider setting DIV [4:0]	Charge Pump setting CHP [4:0]	PLL SYNC MODE SYN CEN	PLL O/P VCO Freq. MHz	VCO Freq. Range VCOD [1:0]	BY-PASS PLL PLL_BYP	PLL O/P Divide Factor	B_CLK_SEL [1:0]	Ts[3] (µs)
31.25	10.0[1]	25	10111	10001	0	250	00	0	8	11	30
32.5	10.0[1]	13	01011	01001	0	130	00	0	4	10	44
35.0	10.0[1]	7	00101	01001	0	70	01	0	2	01	44
37.5	10.0[1]	15	01101	01010	0	150	00	0	4	10	41
40.0[4]	20.0[2]	4	00010	00101	0	80	01	0	2	01	67
42.5	10.0[1]	17	01111	01011	0	170	00	0	4	10	38
45.0	10.0[1]	9	00111	00110	0	90	00	0	2	01	58
47.5	10.0[1]	19	10001	01101	0	190	00	0	4	10	35
50.0[4]	10.0[1]	10	01000	01101	0	100	01	0	2	01	35
52.5	10.0[1]	21	10011	01110	0	210	00	0	4	10	33
55.0	10.0[1]	11	01001	00111	0	110	00	0	2	01	52
57.5	10.0[1]	23	10101	01111	0	230	00	0	4	10	32
60.0[4]	20.0[2]	6	00100	00100	0	120	00	0	2	01	81
62.5	10.0[1]	25	10111	10001	0	250	00	0	4	10	30
65.0	10.0[1]	13	01011	01001	0	130	00	0	2	01	44
70.0[4]	10.0[1]	14	01100	01001	0	140	00	0	2	01	44

Table 14.3 Higher UART_CLK frequencies that can be produced from M_CLK – *not recommended for normal operation*

Notes (for table 14.2 and 14.3):

1) When PLL input frequency is 10.0MHz, this is derived by dividing M_CLK by 4, using PLL_IN_SEL[1:0] set to '11'

2) When PLL input frequency is 20.0MHz, this is derived by dividing M_CLK by 2, using PLL_IN_SEL[1:0] set to '10'.

3) Ts = Worst-case settling time.

4) It is important to ensure that UART_CLK maintains a duty cycle as close as possible to 50:50. This can be achieved by deriving direct multiples of 10MHz over multiplying by a factor of 2, and then dividing the PLL output by 2 accordingly.

Table 14.4 below shows some register values for PLL_CNTL and POW_CNTL, to produce some typical UART_CLK frequencies, based on using an RF Front-end M_CLK as a frequency reference. The values already shown in *Table 14.2 on page 142* are used as a basis for the register values shown. Note that the values used in POW_CNTL register assume that:

1) POWER_GOOD disables ALL functions except Real Time Clock when at logic Low (i.e. Bit 15 is set to '1'.
2) Watchdog function is disabled (i.e. bit 14 is set to '0')
3) ALL Wake-up event control bits are de-activated (i.e. bits 13:11 are set to '0')
4) ALL Sleep enable bits are de-activated (i.e. bits 10:8 are set to '0')
5) Processor Crystal Oscillator block is disabled (i.e. bit 1 is set to '0').

UART_CLK O/P Freq. (MHz)	POW_CNTL register value	PLL_CNTL register value	Comments
10.0	0x803C	0x197F	PLL bypassed and disabled PLL_CNTL value = reset value
11.25	0x80D8	0x018E	
12.5	0x8098	0x09C6	
13.75	0x80D8	0x01D2	
15.0	0x8058	0x1202	
16.25	0x80D8	0x0256	
17.5	0x8098	0x0A4A	
18.75	0x80D8	0x029A	
20.0	0x8036	0x197F	PLL bypassed and disabled PLL_CNTL value = reset value
21.25	0x80D8	0x02DE	
22.5	0x8098	0x018E	
23.75	0x80D8	0x0362	
25.0	0x8058	0x09C6	
26.25	0x80D8	0x03A6	
27.5	0x8098	0x01D2	
28.75	0x80D8	0x03EA	
30.0	0x8018	0x1202	

Table 14.4 SCG register values for UART_CLK frequencies produced from M_CLK (from RF Front end)

For information of the structure of the POW_CNTL and PLL_CNTL registers, refer to *Section 14.6 System Clock Generator Registers on page 146.*

If, the UART_CLK signal is to be derived from an external crystal reference via the on-chip Processor Crystal Oscillator (frequency inputs between 10.0MHz and 16.0MHz), *Table 14.5 below* shows the configurations which can be set-up for the PLL in this case. Note that due to the virtually infinite range of crystal frequencies that can be used, precise frequencies are NOT shown *Table 14.5 below*. Remember that any clock multiple can be divided down to any frequency using the "Divide 1/2/4/8" block which is controlled by the B_CLK_SEL[1:0] register bits.

PLL Mult Factor	Desired PLL output frequency	Prog. Divider Control DIV[4:0]	PLL SYNC MODE SYNCEN	Charge Pump setting CHP[4:0]	VCO Freq. Range VCOD[1:0]	Worst case settling time. (µs)
1	10-32MHz	11111	1	00101	11	67
2	10-32MHz	00000	0	01011	11	38
3	10-32MHz	00001	0	10000	11	31
2	20-63MHz	00000	0	00101	10	67
3	20-63MHz	00001	0	01000	10	47
4	20-63MHz	00010	0	01011	10	38
5	20-63MHz	00011	0	01101	10	35
6	20-63MHz	00100	0	10000	10	31
3	40-125MHz	00001	0	00100	01	81
4	40-125MHz	00010	0	00101	01	67
5	40-125MHz	00011	0	00111	01	52
6	40-125MHz	00100	0	01000	01	47
7	40-125MHz	00101	0	01001	01	44

PLL Mult Factor	Desired PLL output frequency	Prog. Divider Control DIV[4:0]	PLL SYNC MODE SYNCEN	Charge Pump setting CHP[4:0]	VCO Freq. Range VCOD[1:0]	Worst case settling time. (µs)
8	40-125MHz	00110	0	01011	01	38
9	40-125MHz	00111	0	01100	01	36
10	40-125MHz	01000	0	01101	01	35
11	40-125MHz	01001	0	01111	01	32
12	40-125MHz	01010	0	10000	01	31

Table 14.5 PLL configuration options with input freq. from Processor Crystal Oscillator … 1

PLL Mult Factor	Desired PLL output frequency	Prog. Divider Control DIV[4:0]	PLL SYNC MODE SYNCEN	Charge Pump setting CHP[4:0]	VCO Freq. Range VCOD[1:0]	Worst case settling time. (µs)
5	80-250MHz	00011	0	00100	00	81
6	80-250MHz	00100	0	00100	00	81
7	80-250MHz	00101	0	00101	00	67
8	80-250MHz	00110	0	00101	00	67
9	80-250MHz	00111	0	00110	00	58
10	80-250MHz	01000	0	00111	00	52
11	80-250MHz	01001	0	00111	00	52
12	80-250MHz	01010	0	01000	00	47
13	80-250MHz	01011	0	01000	00	44
14	80-250MHz	01100	0	01001	00	44
15	80-250MHz	01101	0	01010	00	41
16	80-250MHz	01110	0	01011	00	38
17	80-250MHz	01111	0	01011	00	38
18	80-250MHz	10000	0	01100	00	36
19	80-250MHz	10001	0	01101	00	35
20	80-250MHz	10010	0	01101	00	35
21	80-250MHz	10011	0	01110	00	33
22	80-250MHz	10100	0	01111	00	32
23	80-250MHz	10101	0	01111	00	32
24	80-250MHz	10110	0	10000	00	31
25	80-250MHz	10111	0	10001	00	30

Table 14.5 PLL configuration options with input freq. from Processor Crystal Oscillator … 2

14.5 System Clock Generator Power Consumption issues

The power consumed by the System Clock Generator is dependent on a number of factors:

1) Processor Crystal Oscillator Block; enabled or disabled

2) 40MHz Differential Input Block; enabled or disabled

3) PLL block; enabled or disabled

4) Input frequency and division ratio of the UART_CLK divider block, after the PLL; the higher the input frequency and the higher the output frequency, the more current consumed.

5) Output frequency of PLL; the higher the output frequency, the more current consumed:

i. 240MHz = 6.2mA;

ii. 125MHz = 4.5mA;

iii. 60MHz = 3.4mA;

iv. 30MHz = 2.9mA

As a general rule of thumb, the lower the PLL input frequency, the lower the PLL output frequency and the lower the UART_CLK divider ratio, the lower the power-consumption of the SCG.

14.6 System Clock Generator Registers

The System Clock Generator uses one primary register, and one register that is shared with functions within the Peripheral Control Logic block (PCL). The Power Control (POW_CNTL) register is actually documented in *Section 12.7.1 "PCL Power Control Register - POW_CNTL* - Memory Offset 0x008*" on page 125*, but the System Clock Generator functions, associated with this register are documented here. All registers in *Table 14.6 below* can be accessed as either byte, half-word, or word.

The GP4020 System Clock Generator Base Address is 0x4010 1000.

Address Offset	Register	Direction	Function
0x008	POW_CNTL	Read/Write	Power Control Register *(shared with Peripheral Control Logic)*
0x00A	PLL_CNTL	Read/Write	PLL Control Register

Table 14.6 System Clock Generator Register Map

14.6.1 SCG Power Control Register - POW_CNTL - Memory Offset 0x008

A write to this register stores logic values which set or reset various enable/disable and switching options within the System Clock Generator. A read of this register shows the status of these functions.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:11	-	*Reserved for use in Peripheral Control Logic*	-	-
10	RF_SLEEP	'1' = Disable 40MHz low-level differential input in System Clock Generator, and apply an active High power-down signal to the RF Front-end (via DISCIO (pin 55), if so configured (ref. IO_REV register). Can be re-enabled by a wake-up event (See Note). '0' = no effect	0	R/W
9	PLL_SLEEP	'1' = Disable and reset the PLL in System Clock Generator. Can be re-enabled by a wake-up event (See Note) after a wait period of approx. 183μs (6 * 32kHz clock cycles; determined by the Real Time Clock block). This allows the CLKINB to stabilise. '0' = no effect	0	R/W
8	-	*Reserved for use in Peripheral Control Logic*	-	-

Bit No.	Mnemonic	Description	Reset Value	R/W
7:6	B_CLK_SEL[1:0]	UART_CLK divider block selector. Allows selection of different output division ratios for the UART_CLK signal, to allow small resolution changes in UART_CLK frequency, if required. The divider ratio is set to divide by 1, in the reset condition. ‘00’ = divide by 1 (i.e. through connection) ‘01’ = divide by 2 ‘10’ = divide by 4 ‘11’ = divide by 8	00	R/W
5	PLL_BYP	PLL Bypass connection. Allows input signal to PLL to appear at input to UART_CLK divider block, effectively removing PLL from signal path. The PLL is bypassed in the reset condition. '1' = Enable PLL bypass, remove PLL from the signal path. '0' = Disable PLL bypass, connect PLL.	1	R/W
4:3	PLL_IN_SEL[1:0]	PLL Input (& PLL Bypass) signal selector. Allows either divided down versions of the M_CLK signal (20.0MHz or 10.0MHz) or a signal from the Processor Crystal Oscillator (10.0MHz to 16.0MHz) to be applied to the PLL CLKINB input as a PLL reference signal. M_CLK / 2 is applied to the PLL CLKINB input in the reset condition. '0x' = connect the output from the Processor Crystal Oscillator to PLL CLKINB input '10' = connect M_CLK / 2 (=20MHz) to PLL CLKINB input '11' = connect M_CLK / 4 (=10.0MHz) to PLL CLKINB input For each change of state, the PLL will be disabled for a wait period of approx. 183µs (6 * 32kHz clock cycles; determined by the Real Time Clock block). This allows the CLKINB to stabilise.	10	R/W
2	PLL_PD	PLL Power Down. PLL is Disabled in the reset Condition '1' = disable the PLL immediately. '0' = Enable the PLL after a wait period of approx. 183µs (6 * 32kHz clock cycles; determined by the Real Time Clock block). This allows the CLKINB to stabilise.	1	R/W
1	PRX_EN	Enable Processor Crystal Oscillator block.. '1' = Enable the Processor Crystal Oscillator; start up time in 10ms typical '0' = disable the Processor Crystal Oscillator. Only to be done if UART_CLK is derived directly from M_CLK.	1	R/W
0	RF_PD	Power Down RF Front-end and 40MHz Low Level Differential Block. Blocks are powered Up in the Reset Condition. '1' = disable the Differential Block and apply an active High power-down signal to the RF Front-end (via DISCIO (pin 55 (100-pin package)), if so configured (ref. IO_REV register). Only possible if UART_CLK is derived from the Processor Crystal Oscillator, as M_CLK will be disabled with the RF Front end. '0' = re-enable the Differential Block and apply an active Low power-on signal to the RF Front-end (via DISCIO (pin 55 (100-pin package)), if so configured (ref. IO_REV register).	0	R/W

Table 14.7 POW_CNTL Register

Note: Wake up event: event which reverses the Sleep activation mode. These are explained in *Section 12.5 "Interrupt and Wake-up logic" on page 121.*

14.6.2 SCG PLL Control Register - PLL_CNTL - Memory Offset 0x00A

A write to this register stores logic values which set or reset input control lines to the PLL within the System Clock Generator. A read of this register shows the status of these functions.

Bit No.	Mnemonic	Description	Reset Value	R/W
15	TM2	*Reserved for PLL Test, in UIM_TEST mode only*	*0*	*Note 1*
14	TM1	*Reserved for PLL Test, in UIM_TEST mode only*	*0*	*Note 1*
13	SG1	*Reserved for PLL Test, in UIM_TEST mode only*	*0*	*Note 1*
12:11	VCOD[1:0]	PLL VCO Output Frequency Range selection pin. This input determines which of the 4 frequency ranges are selected. '00' operates the VCO between 80MHz and 250MHz; '01' operates the VCO between 40MHz and 125MHz; '10' operates the VCO between 20MHz and 63MHz; '11' operates the VCO between 10MHz and 32MHz	11	R/W
10:6	CHP[4:0]	PLL Charge Pump setting. Values determined by numbers in *Table 14.2 on page 142, Table 14.3 on page 143 and Table 14.4 on page 144.*	00101	R/W
5:1	DIV[4:0]	PLL Clock Multiplication Factor. Programmable divider programming bits used for setting up the PLL in clock multiplication mode. These inputs are binary weighted to give divider settings from 2 to 25. The binary value of 'n' will give a divider setting of N = n+2. For clock synchronisation (divide by 1) set SYNCEN to a '1' and DIV0-4 all to a '1' (to minimise power consumption).	11111	R/W
0	SYNCEN	PLL Clock Synchronisation Enable. Allows PLL to produce an output frequency at multiplication factor of 1. '1' = enable Clock Synchronisation mode (i.e. Multiplication fixed at 1) '0' = enable Clock Multiplication mode (i.e. Programmable Multiplication between 2 and 25)	1	R/W

Table 14.8 PLL_CNTL Register

Note 1: In 'UIM_test_mode' (selected by TEST (pin 67) = "1", and TESTMODE (pin 74) = "1"), if the address input, chip select and write enable are doing a "write" to the 'PLL_CNTL' register:

- SG1 is controlled directly by bit 13 of the data bus;
- TM1 is controlled directly by bit 14 of the data bus;
- TM2 is controlled directly by bit 15 of the data bus;

These bits [15:13] are NOT latched in the PLL_CNTL register. All the other bits in the PLL_CNTL register, will be updated at the negative edge of UART_CLK, while the write to the PLL_CNTL address continues).

15 1PPS TIMEMARK GENERATOR

15.1 Introduction

The One Pulse Per Second (1PPS) Timemark Generator is nominally used to provide a 1 ms pulse, once every second, which is phase-aligned to Universal Time Co-ordinated (UTC), in conjunction with GPS system software. The Navstar GPS system relies absolutely on accurate timing information using Atomic clocks in the GPS Satellites. It is possible with the 1PPS Timemark Generator to align the 1PPS Timemark output to UTC, in conjunction with GPS receiver software.

The 1PPS Timemark Generator operates with the Raw Timemark Generator, which exists within the 12-channel correlator block. The Raw Timemark signal is generated from a clock (TIC) which is derived from a 10.000MHz TCXO Receiver Clock Reference (RCR) on a GPS receiver, via numerous frequency dividers in the RF Front-end and GP4020 itself. Consequently, without the use of GPS software to align Timemark to TIC, the Raw Timemark is only as accurate as the frequency stability of the TCXO.

Figure 15.1 below shows a block diagram of the whole Timemark function, including the 1PPS Timemark Generator, and the interface to the 12-channel correlator.

The Timemark output can be accessed from TIMEMARK / TIC (pin 69 (100-pin package)) of the GP4020. Timemark is the default signal, but TIC can also be accessed from this pin by setting the TIC_TIME bit (bit 7) in the TIC_RET register.

RAW_TIMEMARK must be activated in order to access the Timemark output. The Raw Timemark generator can run either of two modes:

- **In Armed mode,** a Raw Timemark output pulse will be generated coincident with the rising edge of the next TIC. Figure 15.2 on page 151 shows the relative timing of the signals used in Armed mode, across 2 seconds, when Timemark is trigger once every 10 TICs. Armed-mode is enabled by writing '1' to the ARM_TIMEMARK bit (bit 0) of the TIMEMARK_CONTROL register before Each Timemark is required.

- **In Free-run mode**, a Raw Timemark pulse is produced coincident with the first rising edge of TIC after the mode is enabled, and then on an integer number of 'n' TICs. The ratio 'n' is known as the FREE_RUN_RATIO, and can be set to any value between 1 and 32 by writing any 5-bit value to bits 2 to 6 of TIMEMARK_CONTROL register. Free-run mode is enabled by writing '1' to the FREE_RUN_TIMEMARK bit (bit 1) of the TIMEMARK_CONTROL register.

TIMEMARK Period = (FREE_RUN_RATIO + 1) * TIC Period (Free run mode)

Of the two modes, normally Armed mode is more useful, since software will have more control on which TIC will trigger a Raw Timemark pulse.

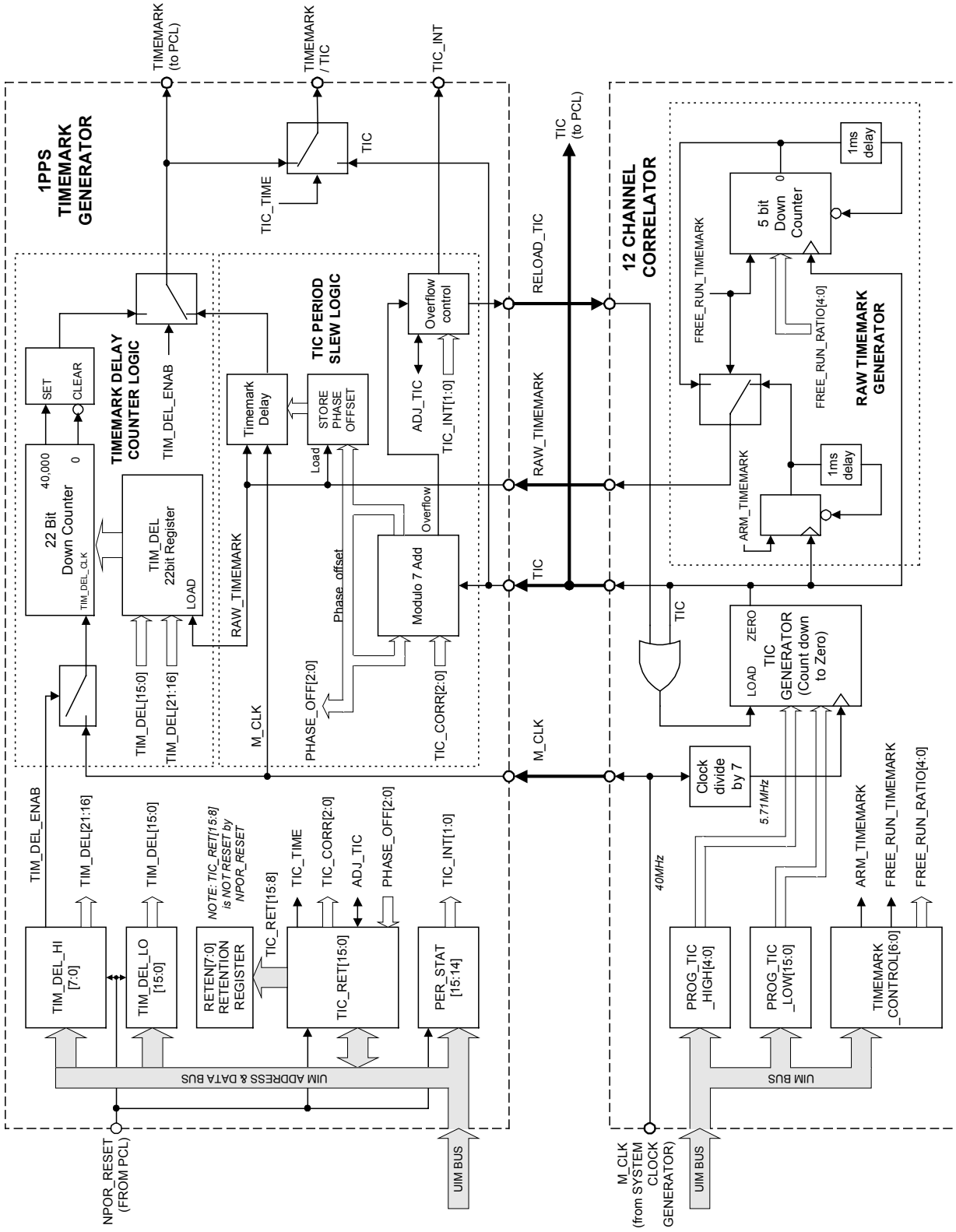


Figure 15.1 1PPS Timemark Generator, with interface to 12-channel correlator block

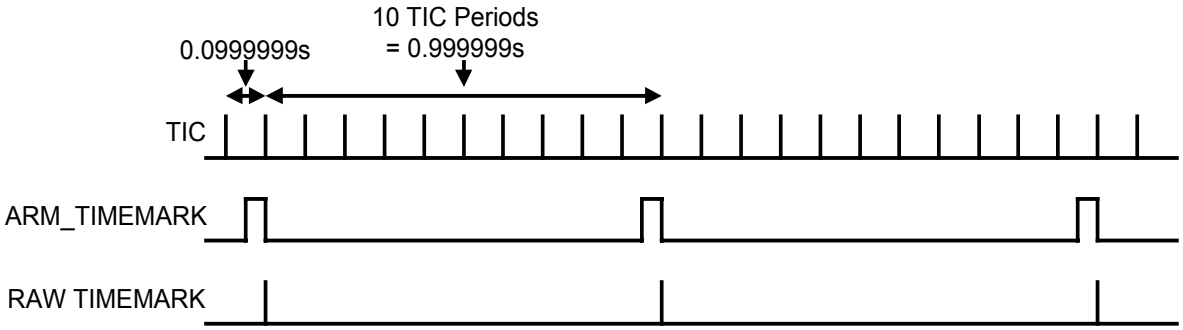


Figure 15.2 Timemark output using ARM_TIMEMARK signal, triggered from software.

The TIC period can only be adjusted in steps of 175ns (7 / 40MHz = 175ns). Therefore, the closest that TIC can be set to 0.1000000s is either 0.0999999s or 0.100000075s. The GP4020 employs some additional fine-resolution adjustment techniques to effectively reduce the TIC period adjustment resolution from 175ns to 25ns; these techniques are indicated later in this section. The TIC period is set to 0.0999999s by default. The period of TIC can be set using the PROG_TIC_HI (5-bits) and PROG_TIC_LO (16-bits) registers, using the following programming routine:

TIC_PERIOD = ((PROG_TIC_HIGH * 65536) + PROG_TIC_LO + 1) * 7 / 40000000

To set TIC from software to be 0.0999999 seconds:

PROG_TIC_HIGH = 0x08,
PROG_TIC_LOW = 0xB823

To set TIC from software to be 0.100000075 seconds:

PROG_TIC_HIGH = 0x08,
PROG_TIC_LOW = 0xB824

If TIMEMARK is set-up to be generated once in every 10 TIC events (where TIC is 0.0999999s), the Timemark period would be 1μs less than 1PPS, and so it would gradually drift away from UTC. It is therefore necessary to continually monitor the relationship between TIC and UTC to effectively keep TIC in phase with UTC. This can be achieved by either slewing the TIC period, or adding a delay after a TIC has occurred.

Figure 15.3 below shows a timing diagram of how Raw Timemark and Timemark are related to TIC, and how small delays are added to align Timemark to UTC. Note the diagram is not to scale, in order to emphasise the delay.

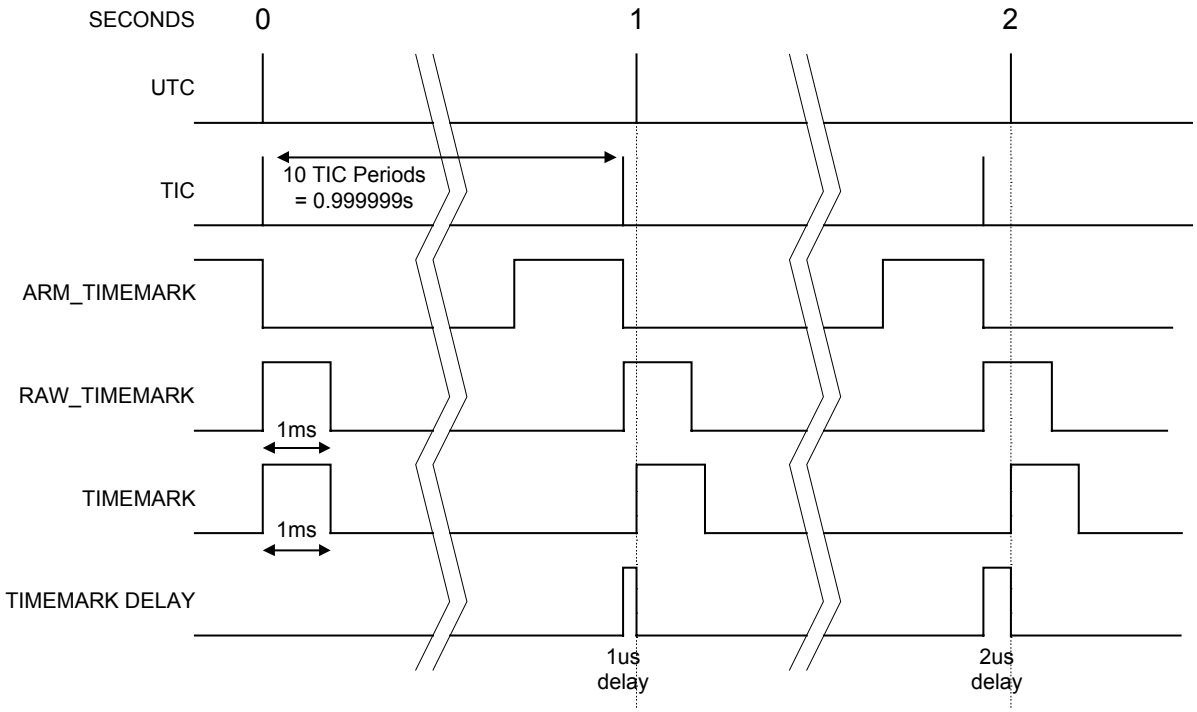


Figure 15.3 Typical timing relationship between UTC, TIC and Timemark, for small Timemark Delay values

The GP4020 employs two separate sets of logic to allow the Timemark output to be aligned to UTC to a resolution of 25ns:

1) TIC period slewing – refer to *Section 15.4 on page 155*
2) Timemark Delay Counter – refer to *Section 15.5 on page 159*

15.2 Issues To Consider When Aligning Timemark To UTC

There are two main issues, which effect the ability to set Timemark to be 1PPS aligned with UTC (Universal Time Co-ordinated).

1) The resolution and value of the TIC period; is TIC stable or does it vary in length.

2) The stability / accuracy of the receiver clock reference, which also impacts the stability / accuracy of TIC. The spec for the 10.000000MHz TCXO in the GPS system is ±2.5ppm, which means that the TIC period is also susceptible to a ±2.5ppm variation – equivalent to the ±250ns for the default TIC period of 0.0999999 secs.

Both of these effects have long-term timing bias implications, unless the software can account for this difference.

To determine a UTC aligned 1PPS TIMEMARK, the receiver will need to be able to PREDICT the corrections needed to a TIC pulse to produce a forthcoming Timemark output pulse. At the time that the next Timemark pulse is required, the GPS receiver needs to know:

i) Absolute value of GPS System Time (i.e. number of GPS seconds which have elapsed in the space of a GPS Week (7days))

ii) The Receiver Clock offset (i.e. the difference in the clock frequency of the Receiver Clock to UTC). This is effectively an offset which shows the time error between a "supposed" one-second period, as timed by the GPS receiver clock, sourced from the receiver TCXO, and an "actual" one-second period as transmitted by the GPS satellites.

iii) The corrections due to ionospheric signal delays and offsets between UTC and GPS System time.

iv) The value of the TIC period is at a given TIC. This can be calculated precisely with respect to UTC if the Receiver Clock Offset is known.

v) The value of UTC at a given TIC, and hence the delay required to be added to a TIC occurrence, in order for Timemark to be aligned to TIC.

vi) Delays due to the RF components and filters (i.e. Group-delay of SAW filters, etc.). These will generally dominate any static error in absolute timing of 1PPS.

vii) Delay due to the propagation delay of the Timemark signal off the chip.

15.3 UTC Error Budget

The following error budget is associated with the generation of the Timemark:

Total Error = Time Transfer Error (TDOP * Ranging Error)
+ Clock Resolution
+ Oscillator Drift Residual Error
+ Computation induced Error
+ Time mark transfer delay through Drivers/cables.
+ Propagation delay in hardware, from antenna to correlator to measurement data sampler

Typical values are:

1. **TDOP (Time Dilution of Precision)**:

   Multiplication factor applied to the Satellite Ranging Error, to give a timing error. TDOP is similar to PDOP, in that the better the geometric distribution of the satellites transmitting the received signals the lower the value.

2. **Ranging error**:

   The ranging error from one satellite is typically about 30m as given in the GPS Navigation message as the User Range Accuracy (URA). This includes errors due to Selective Availability (S/A) and corresponds to a timing error of about 100ns (1 - σ). With multiple satellites, you get an averaging effect but the error in the calculated position also contributes to this.

   In practice, the worst-case Time-Transfer error (TDOP * Ranging Error) is in the region of 250ns and 300ns for a worst case GPS-UTC offset. With DGPS the figure should be well under 50ns and with a surveyed location it should be better again.

3. **Clock Resolution**:

   Calculated as 7.2ns rms. = 25ns / $\sqrt{12}$ (the standard deviation of a uniformly distributed error ranging over 25ns)

4. **Oscillator Drift Residual Error:**

   (a) Due to temperature change on TCXO since last oscillator drift computation: about 50ns, computed with the following assumptions:

   (i) TCXO max. slope is $\pm$1 ppm/°C, typical slope is $\pm$0.2 ppm/°C

   (ii) Temperature max. variation is 5°C/minute.

   (iii) The oscillator drift is computed every second and is at most one second old at UTC time mark.

   For example:

   1ppm/°C x 5°C/min x 1sec = 83ns for a temperature step change
   or 41.5ns (rounded to 50ns) for a linear ramp.

   (b) Due to bias in drift estimation about 50ns max. (rough guess)

TOTAL max. oscillator drift error = (a) + (b).

In practice, the drift is much less than this under typical conditions ≈ 10 to 20ns

5. Computation induced error:

It is assumed that enough significant bits are retained such that this error approximates zero.

6. TIMEMARK transfer delay through drivers/cables:

This bias will be a significant contributor to the total delay figure, due to propagation from the 1PPS Timemark generator through the chip to the output pin and along PCB track or cable to the receiving equipment. The speed of electrical signals through a cable is generally less than speed of light, (approx. 0.25m / ns), but the overall delay due to distance is probably quite small in itself; estimated 2ns.

The main delay will be due to getting the Timemark signal off-chip. With the GP4020 using a CLAOP01L1 output for Timemark, the delay through this port is approx. 11ns with a 10pF external load, and 19ns with a 50pF external load. There is also a residual random delay due to the Timemark output being re-clocked by the M_CLK signal (25ns) when using the Timemark Delay Counter. The latter contributes 25ns/√12 = 7.2ns as a standard deviation.

7. Propagation delays in the hardware:

These are biases that are estimated to be in the range of a few microseconds and are therefore the major contributor to the TIMEMARK synchronisation error. An estimate could be included in the software to improve total accuracy when the total hardware design is complete. Figures already documented within *Section 7.4.11 "Signal Path Delay Introduced by Hardware Signal Processing" on page 61*, suggest that the Hardware delay in getting signals into and sampled by the 12-channel correlator will be in the order of 300ns. (Note that delays after correlation do not affect the TIMEMARK synchronisation. This includes the delays in the accumulators.)

This figure excludes the dominant form of delay due to group-delay effects in filters and circuitry associated with the RF Front-end components. The delay in the Murata SAFCC3542MC00Z SAW filter is of the order of 1200ns and the delays in the other IF filters probably add up to around 500ns giving a total contribution to the bias of around 1700ns.

TOTAL Typical Estimated Bias **≈ 15ns + 300ns + 1700ns**

≈ 2015ns.

TOTAL Typical Estimated Random Error **≈ RMS of (Time Transfer Error**
+ Clock Resolution
+ Oscillator Drift Residual Error
+ Computation induced Error)

≈ $\sqrt{(20^2 + 7^2 + 10^2 + 0 + 7^2)}$ ns.

≈ 25ns.

15.4 Fine-resolution Timemark setting, using TIC period slewing

15.4.1 Functional description

The GP4020 includes some logic within the 1PPS Timemark Generator which allows the TIC period to be specified to a resolution of 1 M_CLK cycle (25ns), without significantly affecting the existing logic in the correlator core.

The default period of TIC is set to be 99999.9μs. Hence it will normally be necessary to add an extra 100ns to the effective period of TIC. This ensures that once every 10 TIC periods, there will be one Timemark output signal that occurs at exactly a one-second period.

A Modulo 7 adder, clocked by TIC from the 12-channel correlator in conjunction with a Timemark delay latch which is clocked by M_CLK (40MHz), can produce "phase-delays" in steps of 25ns into the RAW TIMEMARK signal. The TIC_CORR register can be used to specify the number of M_CLK cycle events (25ns period) to add to each TIC event (up to 175ns). In a typical case, where RAW_TIMEMARK is outputted once every 10 TICs, this gives a maximum adjustment range of delay of 0ns to 1750ns.

At each TIC event, the value in TIC_CORR is added to the existing "Phase_offset". If the result of the addition is greater than or equal to 7, an overflow flag (RELOAD_TIC) is generated. When RELOAD_TIC is set, the load signal to the TIC Generator counter can stay active for an extra TIC cycle. This has a similar result to incrementing the TIC Generator period by 175ns for one TIC period (same effect as incrementing the TIC Generator register "PROG_TIC_LOW" by 1, for one TIC period). The GPS software will normally need to know when the period of TIC is modified by this overflow function, and there are 2 facilities provided for identifying when TIC slewing has occurred / will occur:

1) The interrupt signal TIC_INT is set. This is configured by the PCL block to interrupt the Firefly core, which could then run a simple Interrupt Service Routine, to inform the GPS software that TIC will be 100000.075μs period for one TIC only. (TIC_INT is cleared by a write to CLR_INT (bit 11 in the PER_STAT register).
2) The ADJ_TIC bit (Bit 1 in the TIC_RET register) is set.

At any given time, the current phase_offset can be read from PHASE_OFF[2:0] in TIC_RET register (bits 0 to 2).

If a RAW_TIMEMARK output is produced from the Raw Timemark Generator on the 12-channel correlator block at a given TIC event, the value of "Phase_offset" is latched, and delivered to a 7-stage delay block (Timemark delay), clocked by M_CLK (40MHz). The RAW_TIMEMARK signal is then delayed by between zero and 6 x 25ns using the delay block. Note: the TIMEMARK output will always have an additional half M_CLK cycle delay to allow for re-timing into the Timemark Delay block.

The Timemark delay block is always enabled. However, if a TIMEMARK delay is not required, TIC_CORR can be set to '000', which will give zero M_CLK cycle delays to the RAW_TIMEMARK signal, and hence TIMEMARK = RAW_TIMEMARK.

15.4.2 TIC period slewing Configurations for approximate alignment to 1PPS

This section shows how to configure the 1PPS Timemark generator to produce a Timemark, which is approximately one pulse per second in period, using TIC period slewing. The considerations of aligning time-mark to UTC are NOT covered here, so the Timemark is only approximately 1PPS. The accuracy of the GPS receiver TCXO (which drives the RF Front-end), limits the accuracy of the 1PPS Timemark output. Timemark setting examples, later in this section, illustrate the different scenarios for UTC alignment of Timemark to UTC.

The TIC period slewing correction logic can be configured to work in a number of ways. The ADJ_TIC bit in the TIC_RET register behaves differently for reads and writes, so the following sections use ADJ_TIC_WR (for write to ADJ_TIC) and ADJ_TIC_RD (for read from ADJ_TIC), although they both refer to the same bit in TIC_RET register.

If TIC_INT_EN[1:0] = '00' (in PER_STAT register), the TIC period will not be corrected, since the RELOAD_TIC and TIC_INT signals from the Overflow control logic will be both disabled. However, since the Timemark delay block is always enabled, if TIC_CORR is not set to '000', the RAW_TIMEMARK will be set by (TIC_CORR DIV 7).

TIC_INT_EN[1:0] = '01'. Indicates that the TIC period will automatically be corrected independently of GPS software each time the phase_counter reaches 7, by means of the RELOAD_TIC signal. Each TIC event triggers a new phase_offset calculation. When the new phase_offset calculation is complete, TIC_INT is set (even if TIC period correction is not required). The ADJ_TIC_RD bit indicates if the next TIC period will be extended. Writing to ADJ_TIC has no effect. TIC_INT and ADJ_TIC_RD are used to indicate to the software which TIC periods are being extended.

TIMEMARK will automatically be maintained at approximately 1PPS (TIC_INT and ADJ_TIC_RD are used to indicate to the software which TIC periods are being extended).

TIC_INT_EN[1:0] = 10. Allows the TIC period extension decision to be taken away from Hardware, and allows the decision of whether to extend the TIC period (by means of RELOAD_TIC) to be influenced by software. In this case, TIC_INT is set before a TIC extension is required, as the result of a phase_offset calculation overflow.

If a new phase_offset is calculated, and the result indicates the next TIC period should be extended, both the ADJ_TIC_RD bit and the TIC_INT interrupt signal will be set to '1'. A write ADJ_TIC_WR of '1' will cause the next TIC period to be extended. It will also cause clear ADJ_TIC_RD. (Writing '0' to ADJ_TIC_WR has no effect).

If '1' is not written to ADJ_TIC_WR, the next TIC period will not be extended. Any TIC period can be extended (by writing '1' to ADJ_TIC_WR), irrespective of the state of ADJ_TIC_RD.

To maintain TIMEMARK at approximately 1PPS, '1' should be written to ADJ_TIC_WR after every TIC_INT interrupt.

TIC_INT_EN[1:0] = 11. Allows the TIC period extension decision to be taken away from Hardware, and allows the decision of whether to extend the TIC period (by means of RELOAD_TIC) to be influenced by software. In this case, TIC_INT is set after every phase_offset calculation regardless of whether or not TIC period extension is required (i.e. independent of adder overflow.)

Because of TIC_INT, the ADJ_TIC_RD bit needs to be read to find out if the next TIC period needs to be extended. If ADJ_TIC_RD is set, '1' should be written to ADJ_TIC_WR to extend the next TIC period. (Writing '0' to ADJ_TIC_WR has no effect, if '1' is not written to ADJ_TIC_WR, TIC period will not be extended).

Any TIC period can be extended (by writing '1' to ADJ_TIC_WR), irrespective of the state of ADJ_TIC_RD. To maintain TIMEMARK at approximately 1PPS, at every TIC_INT interrupt, ADJ_TIC_RD should be read, and if it is set, '1' should be written to ADJ_TIC_WR.

The reset condition for the 1PPS Timemark Generator is for TIC_CORR to equal '000', and TIC_INT_EN to equal '00'. If it is left in this condition, the only observable difference to the TIC and TIMEMARK behaviour (between GP4020 and GP2021) is the additional half M_CLK cycle delay on TIMEMARK output.

15.4.3 Timemark setting example 1; TIC period Slewing with No Receiver Clock Offset

It is assumed that the Receiver Clock Reference (i.e. TCXO for RF Front-end) has a zero offset. To automatically correct the timing of 10 cycles of TIC for each Timemark output pulse from 999999μs to 1.000000s, the 1PPS Timemark generator needs to add a 1μs delay to the Timemark output. The default TIC period would need to be set to 99999.9μs, (PROG_TIC_HIGH + PROG_TIC_LOW would be set with 0x08B823). TIC_INT_EN[1:0] would be set to '01', and TIC_CORR would be set to '100' (four M_CLK cycles = 100ns). On the first TIC event, Phase_offset would increase to 4 (which would have no affect on correlator core). If TIMEMARK were generated, it would be delayed by the last Phase_Offset (i.e. 0). On the next TIC event, 'Phase_offset' would change to 1 and RELOAD_TIC would be set (delaying the next TIC event by one cycle of 175ns). This process would continue, as shown in *Table 15.1 below.* TIC Event 0 is assumed phase-aligned to UTC, so the required delay is 0μs.

TIC Event	TIC_ CORR [2:0]	Phase Offset	Offset delay (ns)	Over flow	Next TIC Period (μs)	Cumulated Overflow	Overflow delay (ns)	Total Delay (ns)
0	100	0	0	0	99999.9	0	0	0
1	100	4	100	0	99999.9	0	0	100
2	100	1	25	1	100000.075	1	175	200
3	100	5	125	0	99999.9	1	175	300
4	100	2	50	1	100000.075	2	350	400
5	100	6	150	0	99999.9	2	350	500
6	100	3	75	1	100000.075	3	525	600
7	100	0	0	1	100000.075	4	700	700
8	100	4	100	0	99999.9	4	700	800
9	100	1	25	1	100000.075	5	875	900
10	100	5	125	0	99999.9	5	875	1000

Table 15.1 TIC delay calculations for Timemark using TIC period slew, TIC period with zero error

There are five TIC events out of 10, where the TIC period needs to be adjusted, appearing as coarse 'Phase_clock' corrections inside the correlator core. Timemark output is corrected to the appropriate M_CLK cycle.

15.4.4 Timemark setting example 2; TIC period Slewing with +0.5ppm Receiver Clock Offset

In practice, the TIC period could be upto ±2.5ppm in error due to the Receiver Clock Offset (drift in the receiver TCXO), which equates to approx. ±250ns in a TIC period of 0.0999999s. When a GPS receiver has acquired four or more satellite signals, and has achieved a first fix, the receiver should be able to deduce the error in TIC period due to Receiver Clock offset.

To take the timing example further, assume that the TIC period is in error by -50ns (+0.5ppm. Ppm error is normally equated to frequency, not time, and so the ppm sign is +, NOT -), giving an actual TIC period of 0.09999985s. This means that the correction needed at each TIC to align to UTC, is +150ns. Therefore, at each TIC, a value of '110' (six M_CLK cycles) needs to be added to the phase-offset. The process would occur, shown in *Table 15.2 below*. TIC Event 0 is assumed phase-aligned to UTC, so the required delay is 0μs.

TIC Event	TIC_ CORR [2:0]	Phase Offset	Offset delay (ns)	Over flow	Next TIC Period(μs)	Cumulated Overflow	Overflow delay (ns)	Total delay (ns)
0	110	0	0	0	99999.85	0	0	0
1	110	6	150	0	99999.85	0	0	150
2	110	5	125	1	100000.025	1	175	300
3	110	4	100	1	100000.025	2	350	450
4	110	3	75	1	100000.025	3	525	600
5	110	2	50	1	100000.025	4	700	750
6	110	1	25	1	100000.025	5	875	900
7	110	0	0	1	100000.025	6	1050	1050
8	110	6	150	0	99999.85	6	1050	1200
9	110	5	125	1	100000.025	7	1225	1350
10	110	4	100	1	100000.025	8	1400	1500

Table 15.2 TIC delay calculations for Timemark using TIC period slew, TIC period with +0.5ppm error

There are eight TIC events out of 10, where the TIC period needs to be adjusted, appearing as coarse 'Phase_clock' corrections inside the correlator core. Timemark output is corrected to the appropriate M_CLK cycle.

15.4.5 Timemark setting example 3 - TIC period Slewing with +2.5ppm Receiver Clock Offset

For TIC period errors which are larger than +0.75ppm due to an offset in the Receiver clock, it will be necessary for the receiver to adjust the TIC period to be longer, using the PROG_TIC_HIGH and PROG_TIC_LOW registers.

For example, if the receiver clock offset is noted to be +2.5ppm (-250ns error on TIC period), it is possible to increment the value on PROG_TIC_LOW by 1 from 0xB823 to 0xB824, to set the supposed TIC period to be 0.100000075s. The actual TIC period will be 250ns less than this, at 0.099999825s. With this setting, the correction needed at each TIC to align to UTC, is +175ns. Therefore, at each TIC, a value of '111' (seven M_CLK cycles) needs to be added to the phase-offset. The process would occur, as shown in *Table 15.3 below*. TIC Event 0 is assumed phase-aligned to UTC, so the required delay is 0μs.

TIC Event	TIC_ CORR [2:0]	Phase Offset	Offset delay (ns)	Over flow	Next TIC Period (μs)	Cumulated Overflow	Overflow delay (ns)	Total Delay (ns)
0	111	0	0	0	99999.825	0	0	0
1	111	0	0	1	100000	1	175	175
2	111	0	0	1	100000	2	350	350
3	111	0	0	1	100000	3	525	525
4	111	0	0	1	100000	4	700	700
5	111	0	0	1	100000	5	875	875
6	111	0	0	1	100000	6	1050	1050
7	111	0	0	1	100000	7	1225	1225
8	111	0	0	1	100000	8	1400	1400
9	111	0	0	1	100000	9	1575	1575
10	111	0	0	1	100000	10	1750	1750

Table 15.3 TIC delay calculations for Timemark using TIC period slew, TIC period with +2.5ppm error

All the TIC events will need to adjust the TIC period. If the Receiver Clock offset appears larger still than +2.5ppm, the TIC period will need to be further extended by a further data increment of the PROG_TIC_LOW register.

15.4.6 Timemark setting example 4 - TIC period Slewing with -2.5ppm Receiver Clock Offset

In order to show the opposite end of the Receiver Clock Tolerance limit, this example indicates what to do in order to keep Timemark at 1PPS, when the Receiver Clock Offset means that TIC is longer than 100ms in length. For TIC period errors which are larger than -1.00ppm due to an offset in the Receiver clock, it will be necessary for the receiver to adjust the TIC period to be shorter than the default value, using the PROG_TIC_HIGH and PROG_TIC_LOW registers.

For example, if the receiver clock offset is noted to be -2.5ppm (+250ns error on TIC period), it is possible to decrement the value on PROG_TIC_LOW by 1 from 0xB823 to 0xB822, to set the supposed TIC period to be 0.099999725s. The actual TIC period will be 250ns greater than this, at 0.099999975s. With this setting, the correction needed at each TIC to align to UTC, is +25ns. Therefore, at each TIC, a value of '001' (seven M_CLK cycles) needs to be added to the phase-offset. The process would occur, as shown in *Table 15.4 below*. TIC Event 0 is assumed phase-aligned to UTC, so the required delay is 0μs.

TIC Event	TIC_ CORR [2:0]	Phase Offset	Offset delay (ns)	Over flow	Next TIC Period (μs)	Cumulated Overflow	Overflow delay (ns)	Total Delay (ns)
0	001	0	0	0	99999.975	0	0	0
1	001	1	25	0	99999.975	0	0	25
2	001	2	50	0	99999.975	0	0	50
3	001	3	75	0	99999.975	0	0	75
4	001	4	100	0	99999.975	0	0	100

TIC Event	TIC_ CORR [2:0]	Phase Offset	Offset delay (ns)	Over flow	Next TIC Period (μs)	Cumulated Overflow	Overflow delay (ns)	Total Delay (ns)
5	001	5	125	0	99999.975	0	0	125
6	001	6	150	0	99999.975	0	0	150
7	001	0	0	1	100000.150	1	175	175
8	001	1	25	0	99999.975	1	175	200
9	001	2	50	0	99999.975	1	175	225
10	001	3	75	0	99999.975	1	175	250

Table 15.4 TIC delay calculations for Timemark using TIC period slew, TIC period with -2.5ppm error

All the TIC events will need to adjust the TIC period. If the Receiver Clock offset appears larger still than -2.5ppm, the TIC period will need to be further reduced by a further data decrement of the PROG_TIC_LOW register.

15.5 Fine-resolution Timemark setting, using Timemark Delay Counter

15.5.1 Functional Description

In addition to the TIC slewing logic, described earlier, the GP4020 also includes an alternative method for generating delays of 25ns resolution to the TIC signal, without needing to change the TIC period. Since TIC is used to time most functions within the 12-channel correlator, this can be a very useful method for generating a 1PPS Timemark if the GPS software needs to keep the TIC period constant.

A 22-bit down counter clocked by M_CLK (40MHz) can be used to delay the occurrence of the 1PPS TIMEMARK output from a TIC event. The range of 2^{22} x 25ns, gives a total delay range of upto 104.857575ms (which is slightly larger than the default TIC period). The Timemark output pulse, which is 1ms wide needs to be derived from the counter also and therefore, the last 1ms of the down-count is used to generate the Timemark pulse. This is achieved by setting an output when a down-count reaches 40,000, and then clearing it 1ms later when the count reaches ‘0’.

The use of a 22-bit counter being clocked at 40MHz will consume more power than using the TIC period slewing technique highlighted earlier. However, it will allow 1PPS Timemark to be implemented as an add-on to the GPS function, rather than needing to consider the impact of variable TIC periods on the GPS function.

To enable the Timemark Delay Counter and disable the TIC Slewing logic, set the TIM_DEL_ENAB bit (bit 6) of the TIM_DEL_HI register. This action enables a 40MHz clock to the delay counter, and transfers the counter output to the TIMEMARK output pin, in place of the output from the TIC slewing Timemark delay logic. The value used to set the down-count in the 22-bit down-counter is applied via the TIM_DEL_LO and TIM_DEL_HI registers.

When using the Timemark delay Counter to produce a delay that aligns Timemark to UTC, the sequence of events would be as follows:

1) After a TIC event, the software should decide if RAW_TIMEMARK should be set at the next TIC event, and if so, by how much the TIMEMARK output should be delayed.
2) If TIMEMARK output is to be generated, the software should load the Timemark counter delay with a delay 1ms greater than the desired delay. The value of TIM_DEL_LO MUST be set before programming the TIM_DEL_HI register. Failure to do this will NOT allow the Timemark delay counter to be loaded with the correct value.
3) At the next TIC event, the hardware will set RAW_TIMEMARK.
4) At the first positive edge of M_CLK after RAW_TIMEMARK is set, the Timemark delay counter will be loaded with the value specified in step 2.
5) The Timemark delay counter will then count down to 40000. It will then set the TIMEMARK output register.

6) The Timemark delay counter will continue counting down to 0, at which point the TIMEMARK output register will be cleared and the counter will stop.

7) At the next Timemark event, the same delay value will be used in the Timemark delay counter, unless a new value has been programmed in. The value of TIM_DEL_LO MUST be set before programming the TIM_DEL_HI register. Failure to do this will NOT allow the Timemark delay counter to be loaded with the correct value, and the counter will NOT function.

It should also be noted that the Timemark Delay Counter output is only set if the counter = 40000. If step 2 above loaded a value equivalent to less than 1ms, TIMEMARK would not be set.

This method means that the TIC period can be kept constant. Therefore, the software to run this counter should only have a minimal impact on the GPS function.

15.5.2 Timemark setting example 5 - Timemark Delay Counter with No Receiver Clock Offset

It is assumed that the Receiver Clock Reference (i.e. TCXO for RF Front-end) has a zero offset. To automatically correct the timing of 10 cycles of TIC for each Timemark output pulse from 999999μs, to 1.000000s, the 1PPS Timemark generator needs to add a 1μs delay to the Timemark output. This delay will need to be incremented by 1μs for each Timemark output, until the delay period is greater than or equal to the TIC period. If the calculated delay is greater than or equal to the period of 1 TIC, when the Counter delay can be rolled over, a TIC can be skipped by withholding the RAW_Timemark pulse from the correlator. In this instance, the TIC skip will occur once every 1,000,000 TICs (approx. every 27.7 hours).

The default TIC period would need to be set to 99999.9μs, (PROG_TIC_HIGH + PROG_TIC_LOW would be set with 0x08B823), TIM_DEL_ENAB would be set to '1', and the initial delay required would be 1μs, achieved by 40 M_CLK cycles of 25ns. TIM_DEL_LO is set to be 40,040 or '0x9C68' (40,000 for the 1ms pulse period required), and TIM_DEL_HI is set to '0', although the register should be programmed with a default 0x40 to enable the Timemark delay counter. The process would occur, as shown in *Table 15.5 below*.

Timemark Event (s)	TIC Event	TIC Time (s)	Required delay (μs)	TIM_DEL value	TIM_DEL _LO	TIM_DEL _HI
0	0	0	0	40000	0x9C40	0x40
1	10	0.999999	1	40040	0x9C68	0x40
2	20	1.999998	2	40080	0x9C90	0x40
3	30	2.999997	3	40120	0x9CB8	0x40
100	1000	99.999900	100	44000	0xABE0	0x40
101	1010	100.999899	101	44040	0xAC08	0x40
638	6380	637.999362	638	65520	0xFFF0	0x40
639	6390	638.999361	639	65560	0x0018	0x41
1000	10000	999.999000	1000	80000	0x3880	0x41
1001	10010	1000.998999	1001	80040	0x38A8	0x41
10000	100000	9999.990000	10000	440000	0xB6C0	0x46
10001	100010	10000.989999	10001	440040		
99999	999990	99998.900001	99999	4039960	0xA518	0x7D
(SKIP TIC)	*1000000*	*99999.900000*	*100000*	*4040000*	0xA540	0x7D
100000	1000001	99999.9999999	0.1	40004	0x9C44	0x40
100001	1000011	100000.9999989	1.1	40044	0x9C6C	0x40

Table 15.5 TIC delay calculations for Timemark, using Delay Counter - TIC period with zero error

15.5.3 Timemark setting example 6 - Timemark Delay Counter with +0.5ppm Receiver Clock Offset

In practice, the TIC period could be upto ±2.5ppm in error due to the Receiver Clock Offset (drift in the receiver TCXO), which equates to approx. ±250ns in a TIC period of 0.0999999s. When a GPS receiver has acquired four or more satellite signals, and has achieved a first fix, the receiver should be able to deduce the error in TIC period due to Receiver Clock offset.

To take the timing example further, assume that the TIC period is in error by -50ns (+0.5ppm. PPM error is normally equated to frequency, not time, and so the ppm sign is +, NOT -), giving an actual TIC period of 0.09999985s. This means that the correction needed at each TIC to align to UTC, is +150ns, and consequently a delay of +1.5μs is needed at the first Timemark, +3.0μs at the second time-mark, and the same increment will be added for each Timemark event. If the calculated delay is greater than or equal to the period of 1 TIC, when the Counter delay can be rolled over, and a TIC will be skipped by withholding the TIM_DEL register value. In this instance, the TIC skip will occur once every 660,000 TICs (approx. every 18.5 hours), as shown in *Table 15.6 below.*

Timemark Event (s)	TIC Event	TIC Time (s)	Required delay (μs)	TIM_DEL value	TIM_DEL_LO	TIM_DEL_HI
0	0	0	0	40000	0x9C40	0x40
1	10	0.9999985	1.5	40060	0x9C7C	0x40
2	20	1.999997	3	40120	0x9CB8	0x40
3	30	2.9999955	4.5	40180	0x9CF4	0x40
100	1000	99.999850	150	46000	0xB3B0	0x40
101	1010	100.9998485	151.5	46060	0xB3EC	0x40
425	4250	424.9993625	637.5	65500	0xFFDC	0x40
426	4260	425.999361	639	65560	0x0018	0x41
1000	10000	999.998500	1500	100000	0x86A0	0x41
1001	10010	1000.998499	1501.5	100060	0x86DC	0x41
10000	100000	9999.985000	15000	640000	0xC400	0x49
10001	100010	10000.9849985	15001.5	640060	0xC43C	0x49
66666	666660	66665.900001	99999	4039960	0xA518	0x7D
(SKIP TIC)	*666670*	*66666.8999995*	*100000.5*	*4040020*	*0xA554*	*0x7D*
66667	666671	66666.99999935	0.65	40026	0x9C5A	0x40
66668	666681	66667.99999785	2.15	40086	0x9C96	0x40

Table 15.6 TIC delay calculations for Timemark, using Delay Counter - TIC period with +0.5ppm error

15.5.4 Timemark setting example 7 - Timemark Delay Counter with +2.5ppm Receiver Clock Offset

For TIC period errors which larger than +0.5ppm due to an offset in the Receiver clock, a similar technique for adding delays to the RAW_Timemark signal can be applied, but skipped TICs will occur more frequently.

For example, if the Receiver Clock Offset is noted to be +2.5ppm (-250ns error on TIC period), the actual TIC period is 0.09999965s. This means that the correction needed at each TIC to align to UTC, is +350ns, and consequently a delay of +3.5μs is needed at the first Timemark, +7.0μs at the second time-mark, and the same increment will be added for each Timemark event. If the calculated delay is greater than or equal to the period of 1 TIC, when the Counter delay can be rolled over, and a TIC will be skipped by withholding the TIM_DEL register value. In this instance, the TIC skip will occur once every 285,000 TICs (approx. every 8 hours), as shown in *Table 15.7 below*.

Timemark Event (s)	TIC Event	TIC Time (s)	Required delay (μs)	TIM_DEL value	TIM_DEL _LO	TIM_DEL_ HI
0	0	0	0	40000	0x9C40	0x40
1	10	0.9999965	3.5	40140	0x9CCC	0x40
2	20	1.999993	7	40280	0x9D58	0x40
3	30	2.9999895	10.5	40420	0x9DE4	0x40
100	1000	99.999650	350	54000	0xD2F0	0x40
101	1010	100.9996465	353.5	54140	0xD37C	0x40
182	1820	181.999363	637	65480	0xFFC8	0x40
183	1830	182.9993595	640.5	65620	0x0054	0x41
1000	10000	999.996500	3500	180000	0xBF20	0x42
1001	10010	1000.996497	3503.5	180140	0xBFAC	0x42
10000	100000	9999.965000	35000	1440000	0xF900	0x55
10001	100010	10000.9649965	35003.5	1440140	0xF98C	0x55
28571	285710	28570.9000015	99998.5	4039940	0xA504	0x7D
(SKIP TIC)	*285720*	*28571.899998*	*100002.0*	*4040080*	*0xA590*	*0x7D*
28572	285721	28571.99999765	2.35	40094	0x9C9E	0x40
28573	285731	28572.99999415	5.85	40234	0x9D2A	0x40

Table 15.7 TIC delay calculations for Timemark, using Delay Counter - TIC period with +2.5ppm error

15.5.5 Timemark setting example 8 - Timemark Delay Counter with -2.5ppm Receiver Clock Offset

In order to show the opposite end of the Receiver Clock Offset tolerance limit, this example indicates what to do in order to keep Timemark at 1PPS, when the Receiver Clock Offset means that TIC is longer than 100ms in length.

For TIC period errors which larger than -1.00ppm due to an offset in the Receiver clock, a similar technique for adding delays to the RAW_Timemark signal can be applied, but instead of skipping TICs, additional TICs will need to be introduced. The rate at which additional TICs are added will be proportional to the error in Receiver Clock.

For example, if the Receiver Clock Offset is noted to be -2.5ppm (+250ns error on TIC period), the actual TIC period is 0.10000015s. This means that the correction needed at each TIC to align to UTC, is -150ns, and consequently a delay of -1.5μs is needed at the first Timemark, -3.0μs at the second time-mark, and the same increment will be subtracted for each Timemark event. If the calculated delay decrements to zero, the next Timemark will need to be generated after nine TICs, NOT 10, for that Timemark only. This effectively adds an additional TIC to the Timemark timing. In this instance, the TIC addition will occur once every 660,000 TICs (approx. every 18.5 hours), as shown in *Table 15.8 below.*

Timemark Event (s)	TIC Event	TIC Time (s)	Required delay (µs)	TIM_DEL value	TIM_DEL _LO	TIM_DEL _HI
0	0	0	0	40000	0x9C40	0x40
	(TIC ADD)					
1	9	0.90000135	99998.65	4039946	0xA50A	0x7D
2	19	1.90000285	99997.15	4039886	0xA4CE	0x7D
3	29	2.90000435	99995.65	4039826	0xA492	0x7D
100	999	99.90014985	99850.15	4034006	0x8DD6	0x7D
101	1009	100.90015135	99848.65	4033946	0x8D99	0x7D
1000	9999	999.9014999	98500.15	3980006	0xBAE6	0x7C
1001	10009	1000.90150135	98498.65	3979946	0xBAAA	0x7C
10000	99999	9999.91499985	85000.15	3440006	0x7D86	0x74
10001	100009	10000.91500135	84998.65	3439946	0x7D49	0x74
66666	666659	66665.99999885	1.15	40046	0x9C6E	0x40
	(TIC ADD)					
66667	666668	66666.9000002	99999.8	4039992	0xA538	0x7D
66668	666678	66667.9000017	99998.3	4039932	0xA4FC	0x7D
66669	666688	66668.9000032	99996.8	4039872	0xA4BF	0x7D

Table 15.8 TIC delay calculations for Timemark, using Delay Counter - TIC period with -2.5ppm error

15.6 Data Retention Register

The 1PPS Timemark Generator includes a Data Retention register (TIC_RET[15:8]). This is an 8-bit register, which is NOT reset by any control signals from within the GP4020, and can only be reset by a chip power failure, or by writing a new value to it.

The Data-Retention register can be used as a basic 8-bit UIM test register, to check that the UIM bus accesses are working. This can be done by writing an 8-bit number to it and checking that the same number can be read back.

Additionally, the register can be used by software to indicate a total power-loss to the whole GP4020 chip, including the Real Time Clock section. If a value is written to the register that is then read back some time later after a complete power-failure event has occurred, the values should be significantly different. Hence, this can be used to inform the GPS receiver that the time accumulated in the Real Time Clock is invalid.

15.7 1PPS Timemark Generator Registers

The Timemark Generator uses four registers. These registers are addressable in the same part of the memory map as the Peripheral Control Logic Block - Root address 0x4010 1000.

Address Offset	Register	Direction	Function
0x010	PER_STAT	Read/Write	Used to set interrupts and Resets for the WHOLE GP4020 chip. Primarily used in the Peripheral Control Logic (PCL) block. Bits 14 and 15 used to configure the Tic Period Slew Overflow controller.
0x012	TIC_RET	Read/Write	TIC Retention register, used to configure the TIC Period slewing logic, and access a data retention register.
0x014	TIM_DEL_LO	Read/Write	Timemark Delay Down-Counter setting bits - 15 Least Significant bits
0x016	TIM_DEL_HI	Read/Write	Timemark Delay Down-Counter setting bits - 6 Most Significant bits Also 2 control bits

Table 15.9 1PPS Timemark Generator Register Map

All registers are addressable as 32-bit locations only, but can use sub-memory access. This is particularly useful for the TIM_DEL_HI register, which is only 8-bits wide.

15.7.1 1PPS Timemark STATUS Register - PER_STAT - Memory Offset 0x010

This register is used to control Interrupts and Resets within the GP4020, and most of the controls are handled within the PCL block. The 1PPS Timemark generator uses 2-bits from this register as write bits for the Control of the Timemark Overflow Control block. This is primarily a write-only register, but on reading the register, the settings made by the previous Write can be observed.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:14	TIC_INT_EN[1:0]	Control of the TIC_INT signal interrupt within the Overflow Control Block of the TIC Period slewing logic. '00' = Disable TIC_INT and RELOAD_TIC signals '01' = Enable TIC_INT and RELOAD_TIC signals. TIC_INT and RELOAD_TIC signals generated each time Modulo 7 adder overflows - automatic TIC period extension occurs. ADJ_TIC bit (TIC_RET Register) set also. '10' = Enable TIC_INT and RELOAD_TIC signals. TIC_INT and ADJ_TIC bit (TIC_RET Register) set each time Modulo 7 adder overflows. A Read of ADJ_TIC and subsequent write to ADJ_TIC will set RELOAD_TIC and cause a TIC period extension. If ADJ_TIC not written to, no RELOAD_TIC generated and TIC will not be extended. '11' = Enable TIC_INT and RELOAD_TIC signals. TIC_INT set for every TIC, irrespective of adder overflow state. ADJ_TIC bit (TIC_RET Register) set each time Modulo 7 adder overflows. A Read of ADJ_TIC and subsequent write to ADJ_TIC will set RELOAD_TIC and cause a TIC period extension. If ADJ_TIC not written to, no RELOAD_TIC generated and TIC will not be extended.	00	R/W
13:0		*Reserved for use by Peripheral Control Logic (PCL)*		

Table 15.10 1PPS Timemark PER_STAT Register

15.7.2 1PPS Timemark Generator TIC Retention Register- TIC_RET - Memory Offset 0x012

This register combines control and monitor lines for the 1PPS Timemark Generator TIC period slewing logic, with an 8-bit data retention register.

Note: The Data Retention register bits in the TIC_RET register are NOT reset by any reset event. This can only be cleared by writing '0x00' or powering off GP4020.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:8	RETEN[7:0]	Data Retention Register. An 8bit store location which is NOT reset by any control signals. Could be used to indicate a total power failure in the GPS Receiver. Most Significant Bit = Bit 7	***Not Reset***	R/W
7	TIC_TIME	Toggle signal on TIMEMARK/TIC output pin (pin 69 (100-pin package)): '0' = TIMEMARK output '1' = TIC output	0	R/W
6:4	TIC_CORR[2:0]	TIC Correction (delay) required for each TIC Event, in units of M_CLK cycles (25ns).	000	R/W
3	ADJ_TIC	TIC Period Extension Status / Control bit Read: '1' = TIC period skew Phase_offset calculation has overflowed. Next TIC period will be extended if TIC_INT_EN[1:0] = '01, and should be extended by a write to ADJ_TIC of '1', if TIC_INT_EN[1] = '1']. '0' = next TIC period should NOT be / will NOT be extended Write: '1' = causes next TIC period to be extended (if TIC_INT_EN[1] (in PER_STAT register)) is set. (Also clears set read bit). '0' = No effect.	0	R/W
2:0	PHASE_OFF[2:0]	Current value of Phase offset in Phase-offset calculation.	000	R

Table 15.11 1PPS Timemark TIC_RET Register

15.7.3 1PPS Timemark Generator Delay Counter Register (LSB) - TIM_DEL_LO - Memory Offset 0x014

This register sets the 16 least significant bits for the 22-bit Timemark Delay Counter down-count initialised value, TIM_DEL, in conjunction with TIM_DEL_HI. The value signifies the delay required between a TIC event and a Timemark output pulse, in units of 25ns. The default period for the Timemark output pulse is 1ms, and needs to be added onto the total delay (i.e. add on 40,000) to any delay required in order to give a 1ms pulse. Only values above 40,000 (0x9C40) are valid in this register. All values below 40,000 are ignored.

This is primarily a write-only register, but on reading the register, the settings made by the previous Write can be observed.

Bit No.	Mnemonic	Description	Reset Value	R/W
15:0	TIM_DEL[15:0]	Set LSBs of number of M_CLK clock cycles by which TIC should be delayed before a Timemark Pulse is produced, and completed. Pulse period of 1ms (40,000) should be added onto delay. Most Significant Bit = Bit 15.	0x9C40	R/W

Table 15.12 1PPS Timemark TIM_DEL_LO Register

15.7.4 1PPS Timemark Generator Delay Counter Register (MSB) - TIM_DEL_HI - Memory Offset 0x016

This register sets the six most significant bits for the 22-bit Timemark Delay Counter down-count initialised value, TIM_DEL, in conjunction with TIM_DEL_LO (see above). This is primarily a write-only register, but on reading the register, the settings made by the previous Write can be observed.

Bit No.	Mnemonic	Description	Reset Value	R/W
7	TIM_DEL_TST	*Reserved for Test purposes only.*	0	W
6	TIM_DEL_ENAB	Toggles Timemark offset correction circuitry between: '0' = Enable TIC Period Slewing logic, Disable M_CLK from Timemark Delay Counter logic '1' = Enable Timemark Delay Counter logic, Connect M_CLK to Timemark Delay Counter. Disable TIC Period Slewing logic.	0	R/W
5:0	TIM_DEL[21:16]	6 MSBs of number of M_CLK clock cycles by which TIC should be delayed before a Timemark Pulse is produced, and completed. Most Significant Bit = Bit 5.	0x00	R/W

Table 15.13 1PPS Timemark TIM_DEL_HI Register

16 UP-INTEGRATION MODULE (UIM)

The GP4020 contains the Firefly MF1 core, within which is a Memory Peripheral Controller (MPC) which contains a module called the Up Integration Module (UIM). Within the GP4020, the UIM is used to interface the Firefly MF1 core via the MPC to the following internal memory mapped components:

- 1PPS Timemark Generator
- 12-channel correlator
- Internal Boot ROM
- Internal RAM
- Peripheral Control Logic (PCL)
- Real Time Clock (RTC)
- System Clock Generator (SCG)

The UIM is designed to mimic the behaviour of the MPC external Input / Outputs; the only difference being that the bi-directional sdata bus is split into 2 uni-directional buses.

The UIM is a sophisticated multiplexer and tristate control generator. It provides a link between the MPC port, the SSM broadcast output, the internal Designer logic port and external IO's. There is NO connection to the internal BμILD Bus and consequently the UIM does not contain any internal registers.

Figure 16.1 below shows how the UIM interacts with its environment.

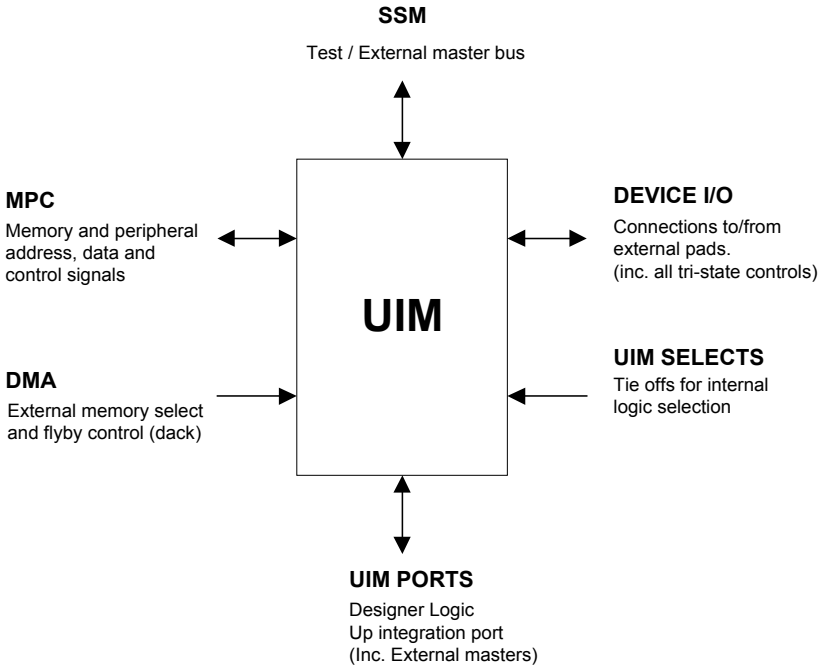


Figure 16.1 Up-Integration Module interfaces

Further details of the function of the Up-Integration Module can be found in *Section 2 of the "Firefly MF1 Core Design Manual" DM5003*, available from Zarlink Semiconductor.

This Page intentionally left blank.

17 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

17.1 Introduction

The GP4020 uses two Universal Asynchronous Receiver Transmitter (UART) modules, which are components that provide industry-standard levels of support for full-duplex asynchronous serial communications, with appropriate mechanisms for software flow control. Neither UART1 nor UART2 support flow control with modem handshake signals (RTS, CTS, DTR, and DSR). Both of the UARTs support DMAC fly-by operation, allowing efficient transmit and receive transfers.

The UARTs support the following features:

- Full duplex operation, independent transmit and receive channels;
- 7- or 8-bit serial data length;
- 1 or 2 stop bits;
- Even, odd or no parity generation;
- Internal selectable baud rate generator, derived from system clock;
- Double buffered transmit and receive channels;
- Software polling to determine channel status;
- Optional interrupt generation on transmit channel becoming empty or receive channel becoming full;
- Detection of parity, overrun, and framing errors on receive channel, with optional interrupt generation;
- Digital input filter to improve noise immunity

The GP4020 incorporates a standard UART as part of the Firefly MF1 Core and an additional UART that is similarly configured, except for the following differences:

1) UART1 is clocked by the Firefly BμILD_CLK, which can be disabled by the F_SLEEP function (refer to *Section 12.5 "Interrupt and Wake-up logic" on page 121*, for more information)

2) UART2 is clocked by the UART_CLK, which is enabled whilst BμILD_CLK is disabled. This allows UART2 to remain active, while the Firefly Core is "asleep".

3) UART2 produces a Data-received interrupt signal, UART_INT, which can be used as a wake-up trigger for the Wake-up and Interrupt Logic in the Peripheral Control Logic. UART_INT is derived from the UART2 "Receive Interrupt" signal, which can be enabled by setting bit 4 of the UART2 Serial Control Register.

The UARTs provide 0V - 3.3V logic levels (not standard RS232 levels). If the UARTs are required to communicate across RS232 lines, and so will need to interface to RS232 lines via inverting level-shifter components.

17.2 Baud Rate Generation

The UART transmit and receive channels are clocked from a single, locally derived clock (referred to below as the **internal clock**), whose period is determined by the reference clock, and the value programmed into the baud rate register. The input clock is provided by:

- BμILD_CLK for UART1
- UART_CLK for UART2

Whilst these are the same frequency, the BμILD_CLK can be disabled by using the F_SLEEP facility (refer to *Section 12.5 "Interrupt and Wake-up logic" on page 121*, for more information).

In both UARTs, the clock pre-scaler, 16 bits long, is configured to generate a reference clock of period **'Sclk** divided by 1, 2, 4, 8, 16, 32, 64, 128, …. upto 32768', as specified by one of the 16 combinations of value [0000 to 1111] programmed into the UART Division Select (MR[7:4]) register.

The Baud Rate Register is used to divide the reference clock period within the range 1 (BRR = 0) to 256 (BRR = 255) to allow approximation to the desired baud rate. The required baud rate register value may be calculated according to the following formula:

BRR = ((Reference Clock) / (16 x Baud Rate)) - 1

The clock is not applied to the transmit or receive channels if neither channel is enabled or currently active. Similarly, the system clock is not applied to the modem control section when the modem enable bit of the control register is reset.

Note that although the Firefly MF1 Core Design Manual (DM5003) mentions modem handshaking signals and an external clock source, 'ueclk', for the UART, these options have NOT been made available on the GP4020. The Baud rate can only be set-up from the BμILD_CLK for UART1, and the UART_CLK for UART2.

17.2.1 Example Baud Rates

The Baud-rate needs to be only approximately correct to allow reliable transmission of data to and from a UART. With this in mind, the baud-rates that can be achieved will be influenced directly by the frequency that the GP4020 System Clock Generator will be asked to run at. There will always be an error in the baud-rate, but this can be minimised by selecting the correct combination of values.

Table 17.1 below through to Table 17.11 on page 174 show the settings of the UART Division select and Baud Rate Register to achieve given baud rates with various values of UART_CLK (BμILD_CLK) frequency. The UART_CLK is based on values that can be produced by the PLL in the System Clock Generator, using the 40MHz M_CLK from an RF front-end as the system clock reference. If any other value of UART_CLK frequency is used (e.g. from a crystal via the Processor Crystal Oscillator), then the value of BRR and Reference Division selection will need to be re-calculated.

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	2.5	129.20833	129	1201.923	-0.16
2400	4	5	129.20833	129	2403.846	-0.16
4800	2	10	129.20833	129	4807.692	-0.16
9600	1	20	129.20833	129	9615.385	-0.16
19200	1	20	64.104167	64	19230.77	-0.16
38400	1	20	31.552083	32	37878.79	1.357
57600	1	20	20.701389	21	56818.18	1.357
76800	1	20	15.276042	15	78125	-1.725
115200	1	20	9.8506944	10	113636.4	1.357

Table 17.1 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 20MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	2.65625	137.34635	137	1203.012	-0.251
2400	4	5.3125	137.34635	137	2406.024	-0.251
4800	2	10.625	137.34635	137	4812.047	-0.251
9600	1	21.25	137.34635	137	9624.094	-0.251
19200	1	21.25	68.173177	68	19248.19	-0.251
38400	1	21.25	33.586589	34	37946.43	1.181
57600	1	21.25	22.057726	22	57744.57	-0.251
76800	1	21.25	16.293294	16	78125	-1.725
115200	1	21.25	10.528863	11	110677.1	3.926

Table 17.2 UART baud-rate settings with UART_CLK (BµILD_CLK) frequency = 21.25MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	2.8125	145.48438	145	1203.981	-0.332
2400	4	5.625	145.48438	145	2407.962	-0.332
4800	2	11.25	145.48438	145	4815.925	-0.332
9600	1	22.5	145.48438	145	9631.849	-0.332
19200	1	22.5	72.242188	72	19263.7	-0.332
38400	1	22.5	35.621094	36	38006.76	1.024
57600	1	22.5	23.414063	23	58593.75	-1.725
76800	1	22.5	17.310547	17	78125	-1.725
115200	1	22.5	11.207031	11	117187.5	-1.725

Table 17.3 UART baud-rate settings with UART_CLK (BµILD_CLK) frequency = 22.5MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	2.96875	153.6224	154	1197.077	0.244
2400	4	5.9375	153.6224	154	2394.153	0.244
4800	2	11.875	153.6224	154	4788.306	0.244
9600	1	23.75	153.6224	154	9576.613	0.244
19200	1	23.75	76.311198	76	19277.6	-0.404
38400	1	23.75	37.655599	38	38060.9	0.883
57600	1	23.75	24.770399	25	57091.35	0.883
76800	1	23.75	18.327799	18	78125	-1.725
115200	1	23.75	11.8852	12	114182.7	0.883

Table 17.4 UART baud-rate settings with UART_CLK (BµILD_CLK) frequency = 23.75MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	3.125	161.76042	162	1198.236	0.147
2400	4	6.25	161.76042	162	2396.472	0.147
4800	2	12.5	161.76042	162	4792.945	0.147
9600	1	25	161.76042	162	9585.89	0.147
19200	1	25	80.380208	80	19290.12	-0.469
38400	1	25	39.690104	40	38109.76	0.756
57600	1	25	26.126736	26	57870.37	-0.469
76800	1	25	19.345052	19	78125	-1.725
115200	1	25	12.563368	13	111607.1	3.119

Table 17.5 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 25MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	3.28125	169.89844	170	1199.287	0.059
2400	4	6.5625	169.89844	170	2398.575	0.059
4800	2	13.125	169.89844	170	4797.149	0.059
9600	1	26.25	169.89844	170	9594.298	0.059
19200	1	26.25	84.449219	84	19301.47	-0.528
38400	1	26.25	41.724609	42	38154.07	0.64
57600	1	26.25	27.483073	27	58593.75	-1.725
76800	1	26.25	20.362305	20	78125	-1.725
115200	1	26.25	13.241536	13	117187.5	-1.725

Table 17.6 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 26.25MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	3.4375	178.03646	178	1200.244	-0.02
2400	4	6.875	178.03646	178	2400.489	-0.02
4800	2	13.75	178.03646	178	4800.978	-0.02
9600	1	27.5	178.03646	178	9601.955	-0.02
19200	1	27.5	88.518229	89	19097.22	0.535
38400	1	27.5	43.759115	44	38194.44	0.535
57600	1	27.5	28.83941	29	57291.67	0.535
76800	1	27.5	21.379557	21	78125	-1.725
115200	1	27.5	13.919705	14	114583.3	0.535

Table 17.7 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 27.5MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	3.59375	186.17448	186	1201.12	-0.093
2400	4	7.1875	186.17448	186	2402.239	-0.093
4800	2	14.375	186.17448	186	4804.479	-0.093
9600	1	28.75	186.17448	186	9608.957	-0.093
19200	1	28.75	92.58724	93	19115.69	0.439
38400	1	28.75	45.79362	46	38231.38	0.439
57600	1	28.75	30.195747	30	57963.71	-0.631
76800	1	28.75	22.39681	22	78125	-1.725
115200	1	28.75	14.597873	15	112304.7	2.513

Table 17.8 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 28.75MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	3.75	194.3125	194	1201.923	-0.16
2400	4	7.5	194.3125	194	2403.846	-0.16
4800	2	15	194.3125	194	4807.692	-0.16
9600	1	30	194.3125	194	9615.385	-0.16
19200	1	30	96.65625	97	19132.65	0.351
38400	1	30	47.828125	48	38265.31	0.351
57600	1	30	31.552083	32	56818.18	1.357
76800	1	30	23.414063	23	78125	-1.725
115200	1	30	15.276042	15	117187.5	-1.725

Table 17.9 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 30MHz

Note: *The data supplied in Table 17.10 and Table 17.11 below is provided **for information only**. Operating the GP4020 UART_CLK at a frequency above 30MHz is NOT recommended for normal operation.*

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	4.0625	210.58854	211	1197.671	0.194
2400	4	8.125	210.58854	211	2395.342	0.194
4800	2	16.25	210.58854	211	4790.684	0.194
9600	1	32.5	210.58854	211	9581.368	0.194
19200	1	32.5	104.79427	105	19162.74	0.194
38400	1	32.5	51.897135	52	38325.47	0.194
57600	1	32.5	34.264757	34	58035.71	-0.756
76800	1	32.5	25.448568	25	78125	-1.725
115200	1	32.5	16.632378	17	112847.2	2.042

Table 17.10 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 32.5MHz

Baud Rate Required	Division Select Value	Reference Clock (MHz)	Baud Rate Ratio	Prog. BRR Value	Programmed Baud Rate	Baud Rate Error (%)
1200	8	4.375	226.86458	227	1199.287	0.059
2400	4	8.75	226.86458	227	2398.575	0.059
4800	2	17.5	226.86458	227	4797.149	0.059
9600	1	35	226.86458	227	9594.298	0.059
19200	1	35	112.93229	113	19188.6	0.059
38400	1	35	55.966146	56	38377.19	0.059
57600	1	35	36.977431	37	57565.79	0.059
76800	1	35	27.483073	27	78125	-1.725
115200	1	35	17.988715	18	115131.6	0.059

Table 17.11 UART baud-rate settings with UART_CLK (BμILD_CLK) frequency = 35MHz

17.3 Connections to the BμILD bus and the Firefly MF1 Core

The GP4020 UARTs are configured such that UART1 is a standard Firefly MF1 component, and UART2 is also connected to Firefly via the external BμILD bus.

The Base Addresses of the GP4020 UARTs are as follows:

- UART1 = 0xE001 8000,
- UART2 = 0xE001 9000.

Each UART produces three interrupt lines to the Firefly Interrupt Controller (INTC):

UART1 Tx / Rx error	- INTC channel 14
UART1 Receive Buffer Full	- INTC channel 15
UART1 Transmit Buffer Full	- INTC channel 16
UART2 Tx / Rx error	- INTC channel 26
UART2 Receive Buffer Full	- INTC channel 27
UART2 Transmit Buffer Full	- INTC channel 28

Since INTC channel 0 has highest priority, this means that UART1 interrupts will have a higher priority than UART2 interrupts.

Further details of the function and programming of the GP4020 Universal Asynchronous Receiver Transmitter (UART) can be found in *Section 4 of the "Firefly MF1 Core Design Manual" DM5003*, available from Zarlink Semiconductor.

This Page intentionally left Blank

18 WATCHDOG TIMER (WDOG)

The function of the Watchdog Timer block [WDOG] is to detect hardware or run-time software errors. It performs this function by requiring the processor to write to one of its registers periodically. Should this not occur, the Watchdog will time-out and reset the system. This ensures that hardware/software lock-ups are recoverable. A watchdog timer exists on the BμILD bus to verify that the ARM® code is always in a known state. A value must be written to the timer within a programmed period to avoid an interrupt being sent to the ARM7TDMI processor. The Watchdog timer is disabled by default by a chip reset, but can be enabled by setting WATCH_EN (Bit 14 of the POW_CTRL register in the PCL block) to a Logic '1'. Either a time-out failure signal is triggered by the 8-bit timer expiring, or a register bit set to raise the signal. In either case this signal is intended to shut off the GP4020 in order to reset it.

The watchdog timer contains a 32-bit counter and an 8-bit counter. Each of the counters has programmable load values. A 32-bit (primary) counter counts down until either it reaches zero or a fixed value is written to the watchdog control register. On expiration, an interrupt is sent to the ARM7TDMI to notify it of Watchdog expiration and the secondary 8-bit counter begins counting down.

If the watchdog is not written to by the time the 8-bit counter counts down, the ASIC is reset. The primary counter is clocked directly from the Firefly MF1 Subsystem clock (UART_CLK). The secondary counter is clocked by UART_CLK after passing through a divide-by-sixteen pre-scaler.

The watchdog timer generates periodic interrupts (WATCH_INT) to the ARM7TDMI processor, which can also be monitored within the Peripheral Control Logic block, to allow system wake-up to occur at the time of an interrupt. Should the processor not respond to the interrupt within a certain time a secondary counter times out and triggers the Peripheral Control Logic to provide a system-wide reset via the NPOR_RESET and NRESET lines.

To respond, the processor needs to write a specific, predefined value into the restart register. This "key" is necessary to guard against errant software accidentally restarting the watchdog, since it is extremely unlikely that it would write the correct 32-bit number into the watchdog restart register. The WDOG output is reset via the external NSRESET signal.

18.1 Design Features

The main features of this watchdog timer are:

- “Key” mechanism for restarting the watchdog prevents accidental restarts
- Adjustable interrupt frequency
- Adjustable time-out delay

A Block diagram of the GP4020 Watchdog is shown in *Figure 18.1 below*.

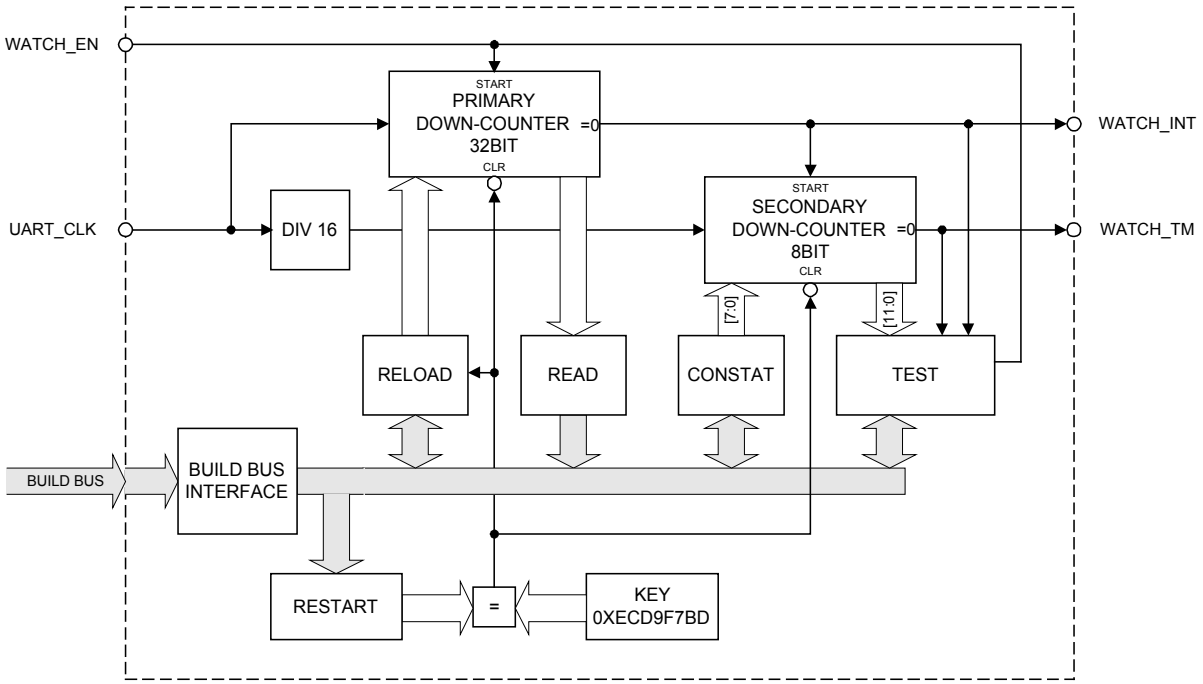


Figure 18.1 Watchdog Block Diagram

18.2 Operational Description

The watchdog consists of two counters; the primary, which is 32-bits long, and the secondary, which is 8-bits long. Both counters are clocked off the system clock, UART_CLOCK; the primary counter is clocked directly, and the secondary counter via a divide-by-sixteen pre-scaler.

18.2.1 Start-up behaviour

After system reset, the enable signal 'WATCH_EN' controls the Watchdog start-up behaviour. In the GP4020, the WATCH_EN signal is provided by bit 14 of the POW_CNTL register in the Peripheral Control Logic block, which has a reset value of '0'.

To enable the Watchdog, either of the following techniques can be used:

- Set Bit 14 of the PCL POW_CNTL register to "1";
- Write the "Restart key" to the Watchdog RESTART register (see below for details);

Once the watchdog is enabled, it cannot be disabled without resetting the GP4020. However, the watchdog can be held off if the Watchdog RESTART key will need to be written to the RESTART register.

18.2.2 Timer Operation and Watchdog Restart Key

When enabled, the primary counter counts down from its 32-bit reload value towards zero. On reaching zero, the Watchdog will generate an interrupt to the ARM7TDMI processor. The interrupt can be masked out by using the MSK bit in the watchdog control register.

It then starts the secondary counter counting down to zero and sets the overflow (OVF) flag in the control register. If the processor fails to restart the watchdog before the secondary counter reaches zero, it will generate a time-out signal, which causes a system reset.

To restart the watchdog counter, a specific 32-bit value (=0xECD9F7BD; the RESTART key) must be programmed into the watchdog restart register. This 32-bit "key" activates the restart mechanism, which performs the following actions:

1) Loads the watchdog primary counter from the reload register
2) Resets the OVF (overflow) flag in the control register
3) Restarts the primary counter
4) Reloads and disables the secondary time-out counter
5) Removes the time-out signal

The reload register enables the interrupt period to be varied in software. Changes to the reload register value will only be reflected in the main watchdog counter after the next software reload.

18.2.3 Watchdog Timer ranges

The input clock to the Watchdog is UART_CLK, which is essentially the Firefly System Clock. Typically, the default UART_CLK frequency is 20MHz, unless the System Clock Generator sets this to another value.

The Watchdog Primary Counter, used to define the Watchdog Interrupt period, has a 32-bit range, and hence has a maximum value of 2^{32} ~= 4295million. With a 20MHz clock this gives a maximum interrupt period of:

4295000000 / 20000000 = 215 seconds

The Watchdog Secondary counter, used to define the Watchdog Time-out period, has an 8-bit range, and hence has a maximum value of 2^8 = 256. The clock into the Secondary counter is the UART_CLK input divided by 16, which gives an effective clock of 1.25MHz when UART_CLK is 20MHz. With UART_CLK = 20MHz, the maximum Watchdog Time-out period is:

256 / 1250000 = 204.8μs.

In a GPS system using the GP4020, it is recommended that the Interrupt service routine which is generated in software allows for the fact that the Watchdog Interrupt should be serviced quickly, in order to avoid a complete chip reset.

18.3 Watchdog Register Map

The GP4020 Watchdog base address is 0xE0004000.

The addresses in the Register Map are specified as offsets from the Watchdog base address.

Address Offset	Register	Direction	Function
0x000	CONSTAT	Read/Write	Control and status bits + Secondary Counter Start Value (Time-out period).
0x004	RELOAD	Read/Write	Primary Counter Start Value (interrupt frequency).
0x008	READ	Read	Read current Primary Counter Value. *Only accessible in TEST mode.*
0x00C	RESTART	Write	Restart "key" input. Key value = 0xECD9F7BD.
0x010	TEST	Read/Write	Access test signals + Read current Secondary Counter Value. Only accessible in TEST mode.
0x014.....0xFFF			*Reserved*

Table 18.1 Watchdog Register Map

TEST mode can be activated only by setting GP4020 'TEST' input (pin 67 (100-pin package)) to Logic '1', and 'TESTMODE' (pin 74 (100-pin package)) is set to Logic '0'.

All Watchdog Registers are 32-bits wide.

18.3.1 Watchdog Control / Status Register - CONSTAT - Memory Offset 0x000

The control register is 32-bits wide, with unused bits defined as zero. Attempts to access the register as a byte or a half-word will cause a bus error exception.

Bit No.	Mnemonic	Description	Reset Value	R/W
31:20		*Reserved*	0	-
19	EN	Mask that controls whether the zero count state of the watchdog generates an interrupt to the ARM7TDMI processor. '1' = Generate interrupt '0' = Mask Disabled	0	R/W
18	OVF	Read-only: Overflow - set whenever the watchdog time-out (secondary) counter reaches zero. OVF can only be reset by restarting the watchdog timer.	0	R
17		*Reserved*	0	-
16	RUN	Read-only: Run Status - indicates whether the Watchdog counters are enabled and running at the moment of access. '1' = Watchdog running '0' = Watchdog NOT running	0	R
15:8		*Reserved*	0x00	-
7:0	TIME_DELAY	Time-out delay used by secondary (Time-out) counter. Value equates to number of (UART_CLK/16) cycles required in the delay. The secondary counter counts down from this value to zero and then generates a system reset unless the WDOG is re-started. Most Significant Bit: Bit 7	0xFF	R/W

Table 18.2 Watchdog CONSTAT Register

18.3.2 Watchdog Primary Counter Reload Register - RELOAD - Memory Offset 0x004

The PR_RELOAD value is loaded into the Primary Interrupt Counter each time the RESTART Key (=0xECD9F7BD) is sent to the RESTART register. It signifies the time interval between WATCH_INT interrupt events:

Watchdog Interrupt period = PR_RELOAD / UART_CLK frequency

Bit No.	Mnemonic	Description	Reset Value	R/W
31: 0	*PR_RELOAD*	The WDOG primary counter is reloaded with this value following a WDOG re-start or a system reset. After reloading the WDOG counts from this value down towards zero.	0xFFFFFFFF	R/W

Table 18.3 Watchdog RELOAD Register

18.3.3 Watchdog Primary Counter Read Register - READ - Memory Offset 0x008

Bit No.	Mnemonic	Description	Reset Value	R/W
31: 0	*PC_READ*	Current primary counter read value. *Accessible in TEST mode only.*	0xFFFFFFFF	R

Table 18.4 Watchdog READ Register

18.3.4 Watchdog Restart Register - RESTART - Memory Offset 0x00C

Bit No.	Mnemonic	Description	Reset Value	R/W
31: 0	*RESTART_KEY*	Restart Key. A write to this register with the correct key value, 0xECD9F7BD, restarts the WDOG primary counter.	0x00000000	W

Table 18.5 Watchdog RESTART Register

18.3.5 Watchdog TEST Register - TEST - Memory Offset 0x010

This register is only accessible when the GP4020 is put into TEST mode (i.e. 'TEST' (pin 67 (100-pin package)) is set to Logic '1', and 'TESTMODE' (pin 74 (100-pin package)) is set to Logic '0').

Bit No.	Mnemonic	Description	Reset Value	R/W
31:21		*Reserved*		
20	T_RST	Replace reset signal 'slave_reset' in TEST mode.	0	R
19	T_CLK	Replace clock signal 'wd_clk' in TEST mode.	0	R
18	T_EN	Replace enable signal 'wd_en' in TEST mode.	0	R
17	T_INT	Current interrupt signal 'wd_int'.	0	R
16	T_TOUT	Current time-out signal 'wd_tout'	0	R
15:12		*Reserved*		
11:0	SC_READ	Current secondary counter read value	0	R

Table 18.6 Watchdog TEST Register

The relationship between the test signals and the external signals are shown in *Table 18.7 below*.

Test mode	External signal	Test signal	IO	Origin	Destination
0	wd_clk	--	I	extern	Primary/Secondary Counters
1	--	t_clk	I	test register	Primary/Secondary Counters
0	wd_en	--	I	extern	Primary Counter
1	--	t_en	I	test register	Primary Counter
0	wd_int	--	O	Primary Counter	extern
1	--	t_int	O	Primary Counter	test register
0	wd_tout	--	O	Secondary Counter	extern
1	--	t_tout	O	Secondary Counter	test register

Table 18.7 Watchdog test signals

19 ADDRESS MAPS

19.1 GP4020 System Address Map

The GP4020 has the Address Map as shown in *Table 19.1 below*.

ADDRESS RANGE	FUNCTION	MPC Area	NOTES
0x0000 0000 - 0x000F FFFF	Internal Boot ROM or External Boot ROM via NSCS[0]	1	1
0x0010 0000 - 0x1FFF FFFF	*Not Used (Int./Ext. Boot ROM reflected)*		
0x2000 0000 - 0x200F FFFF	External Chip Select 1 via NSCS[1]	2	
0x2010 0000 - 0x3FFF FFFF	*Not Used (CS1 reflected)*		
0x4000 0000 - 0x400F FFFF	External Chip Select 2A via NSCS[2A] *NOTE: Correlator, Peripherals REFLECTED every 0x2000*	3	2
0x4010 0000 - 0x4010 0FFF	12-Channel Correlator	3	2
0x4010 1000 - 0x4010 1FFF	Peripheral Control Logic, Real Time Clock, System Clock Generator, 1PPS Timemark Generator	3	2
0x4010 2000 - 0x5FFF FFFF	*Not Used (CS2, Correlator, Peripherals reflected)*		
0x6000 0000 - 0x6000 1FFF	Internal SRAM - 2k x 32-bit	4	
0x6000 2000 - 0x7FFF FFFF	*Not Used (CS3 reflected)*		
0x8000 0000 - 0xDFFF FFFF	*Reserved*		
0xE000 0000 - 0xE00F FFFF	Firefly MF1 BμILD bus modules		
0xE010 0000 - 0xFFFF FFFF	*Reserved*		

Table 19.1 GP4020 System Address Map

19.1.1 GP4020 System Address Map Notes

1) Internal or External ROM is selected by the state of MULTI_FNIO (pin 54 (100-pin package)) during chip reset via NSRESET (pin 75 (100-pin package)) Low or Watchdog time-out. External ROM selected if MULTI_FNIO set low. Internal ROM selected by MULTI_FNIO set High.

2) External Chip Select 2A (NSCS[2A]), the internal 12-channel correlator and GP4020 Peripherals share the same wait-state characteristics as defined within Memory Peripheral Controller.

   The NSCS[2A] chip-select line is active for ALL memory accesses in Memory Area 3 (i.e. ALL memory space between 0x4000 0000 and 0x5FFF FFFF, including the internal logic blocks as defined below:

   - (System Address Bit 20 = 1 AND System Address Bit 12 = 0) selects the GP4020 Correlator.
   - (System Address Bit 20 = 1 AND System Address Bit 12 = 1) selects the GP4020 Peripherals (Peripheral Control Logic, Real Time Clock, System Clock Generator, and 1PPS Timemark Generator).

   An access to an external area WILL NOT be reflected into the internal logic blocks, but an internal area access WILL be reflected externally.

   What this means is that the Address Space occupied between 0x4010 0000 and 0x4010 1FFF (i.e. 0x0000 to 0x1FFF) will be reflected through the whole Area 3 Address space once every 0x2000.

   The limitation indicated in NSCS[2A] can be bypassed by undertaking either of the following fixes in Hardware:

a) Gate NSCS[2A] externally with SADD[19] to produce a smaller external address space for NSCS[2A], but without the reflection of the internal logic once every 0x2000. The truth-table shown in *Table 19.2 below*:

NSCS[2A]	SADD[19]	EXTERNAL CHIP SELECT
0	1	0 (ENABLED)
0	0	1 (DISABLED)
1	1	1 (DISABLED)
1	0	1 (DISABLED)

Table 19.2 Truth Table for NSCS[2A] to avoid external reflection of internal accesses, using SADD[19]

This produces the change to the Address-Map, as shown in *Table 19.3 below*:

ADDRESS RANGE	FUNCTION	MPC Area
0x4000 0000 - 0x4007 FFFF	Reserved (12-Channel Correlator, Peripheral Control Logic, Real Time Clock, System Clock Generator, 1PPS Timemark Generator reflected every 0x2000)	3
0x4008 0000 - 0x400F FFFF	External Chip Select 2A via NSCS[2A], gated with SADD[19] = "1"	3
0x4010 0000 - 0x4010 0FFF	12-Channel Correlator	3
0x4010 1000 - 0x4010 1FFF	Peripheral Control Logic, Real Time Clock, System Clock Generator, 1PPS Timemark Generator	3
0x4010 2000 - 0x5FFF FFFF	*Reserved (CS2, Correlator, Peripherals reflected)*	

Table 19.3 GP4020 Memory Area 3 Addressing, with modified NSCS[2A] logic

b) Use a GPIO line from GP4020 to externally gate NSCS[2A]. This will allow the maximum utilisation of the external Memory Area 3 space, but requires software to configure the GPIO line, which could be inefficient. The truth-table shown in *Table 19.4 below* applies:

NSCS[2A]	GPIO	EXTERNAL CHIP SELECT
0	1	0 (ENABLED)
0	0	1 (DISABLED)
1	1	1 (DISABLED)
1	0	1 (DISABLED)

Table 19.4 Truth Table for NSCS[2A] to avoid external reflection of internal accesses, using GPIO line

This produces the following change to the Address-Map, as shown in *Table 19.5 below*:

ADDRESS RANGE	FUNCTION	MPC Area	NOTES
0x4000 0000 - 0x400F FFFF	External Chip Select 2A via NSCS[2A], gated with GPIO = "1", in software.	3	2
0x4010 0000 - 0x4010 0FFF	12-Channel Correlator	3	2
0x4010 1000 - 0x4010 1FFF	Peripheral Control Logic, Real Time Clock, System Clock Generator, 1PPS Timemark Generator	3	2
0x4010 2000 - 0x5FFF FFFF	*Reserved (CS2, Correlator, Peripherals reflected)*		

Table 19.5 GP4020 Memory Area 3 Addressing, with modified NSCS[2A] logic

19.2 GP4020 Firefly MF1 Address Map

The Firefly MF1 BμILD bus modules have the address map as shown in *Table 19.6 below*, in the range 0xE000 0000 to 0xE007 FFFF.

ADDRESS RANGE	FUNCTION
0xE000 2000 - 0xE000 2FFF	System Configuration (including SSM)
0xE000 4000 - 0xE000 4FFF	Watchdog (WDOG)
0xE000 5000 - 0xE000 5FFF	General purpose I/O block (GPIO)
0xE000 6000 - 0xE000 6FFF	Interrupt Controller (INTC)
0xE000 7000 - 0xE000 7FFF	Serial Input / Output block (BSIO)
0xE000 8000 - 0xE000 8FFF	Memory/Peripheral Controller (MPC)
0xE000 C000 - 0xE000 CFFF	DMA Controller (DMAC)
0xE000 E000 - 0xE000 EFFF	Timer 1 (TIC1)
0xE000 F000 - 0xE000 FFFF	Timer 2 (TIC2)
0xE001 8000 - 0xE001 8FFF	UART1
0xE001 9000 - 0xE001 9FFF	UART2
All other areas in the range : 0xE000 0000 - 0xFFFF FFFF	*Reserved*

Table 19.6 Firefly MF1 Address Map

Further details on how to configure all the GP4020 Memory Areas, is described in *Section 11 "MEMORY PERIPHERAL CONTROLLER (MPC)" on page 109.*

Further details on Wait-state Generation can be found in *Section 3.3.3 of the "Firefly MF1 Core Design Manual" (DM5003)*, also available from Zarlink Semiconductor.

This page intentionally left blank.

20 INPUT / OUTPUT PIN CHARACTERISTICS

The GP4020 Electrical Characteristics, which are specific to the GP4020 device are shown in the "GP4020 GPS Baseband Processor Datasheet", DS5134, available from Zarlink Semiconductor.

In this section, the generic characteristics of the Input Output pins, and some timing diagrams of important external interfaces are shown. The data shown here should be used in conjunction with the data in the datasheet specified above.

20.1 Pin Types

The definitions of the type of each pin for the 100-pin GP4020 is shown in *Table 20.1 below*, with some additional notes relating to them.

Pin No.	Pin Name	Pin Type	IP / OP Cell Type	5V Tol.?	Hold Cell?	Pull-up/ -down	Tri-state	Notes
1	SADD[0]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
2	SADD[1]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
3	SADD[2]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
4	SADD[3]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
5	SADD[4]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
6	SADD[5]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
7	GND	PWR	CLALLVM					
8	SADD[6]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
9	SADD[7]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
10	VDD	PWR	CLALLVP					
11	NSCS[0]	IP/OP	CLAIO1HD01N	No	-	None	Yes	1, 2
12	NSCS[1]	OP	CLAOP01N	No	-	None	Yes	1
13	NSCS[2A]	OP	CLAOP01N	No	-	None	Yes	1
14	SADD[19]	OP	CLAOP03N	No	-	None	*Note 3*	3
15	SDATA[0]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
16	SDATA[1]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
17	SDATA[2]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
18	SDATA[3]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
19	GND	PWR	CLALLVM					
20	SDATA[4]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
21	SDATA[5]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
22	VDD	PWR	CLALLVP					
23	SDATA[6]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
24	SDATA[7]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
25	NSOE	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	1, 2
26	NSWE[1]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	1, 2
27	NSWE[0]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	1, 2
28	SDATA[8]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
29	SDATA[9]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
30	VDD	PWR	CLALLVDD					
31	SDATA[10]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
32	SDATA[11]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
33	GND	PWR	CLALLGND					

20: Input / Output pin Characteristics

Pin No.	Pin Name	Pin Type	IP / OP Cell Type	5V Tol.?	Hold Cell?	Pull-up/ -down	Tri-state	Notes
34	SDATA[12]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
35	SDATA[13]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
36	SDATA[14]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
37	SDATA[15]	IP/OP	CLAIO1HD03N	No	Yes	None	Yes	
38	SADD[18]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
39	SADD[17]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
40	SADD[16]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
41	GND	PWR	CLALLGND					
42	SADD[15]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
43	SADD[14]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
44	VDD	PWR	CLALLVDD					
45	SADD[13]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
46	SADD[12]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
47	SADD[11]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	*2, 3*
48	SADD[10]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
49	SADD[9]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
50	SADD[8]	IP/OP	CLAIO1HD03N	No	Yes	None	*Note 3*	2, 3
51	SWAIT	IP	CLAIP1GD	No	No	100k down	-	
52	NSUB	OP	CLAOP03N	No	-	-	Yes	
53	IEXTINT2	IP	CLAIP1NR	No	No	None	-	
54	MULTI_FNIO	IP/OP	CLAIO1NR01N	No	No	None	Yes	
55	DISCIO	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
56	RF_PLL_LOCK	IP	CLAIP1NR	No	No	None	-	
57	A1VDD	PWR	CLPLAN1VP					
58	CLK_T	IP	CLHANPIP to SCLVDSRX					
59	CLK_I	IP	CLHANPIP to SCLVDSRX					
60	GND	PWR	CLALLVM					
61	SIGN0	IP	CLAIP1NR	No	No	None	-	
62	MAG0	IP	CLAIP1NR	No	No	None	-	
63	SAMPCLK	OP	CLAOP01N	No	-	-	No	
64	POWER_GOOD	IP	CLAIP1NR	No	No	None	-	
65	PR_XOUT		CLGOSC1016M					
66	PR_XIN		CLGOSC1016M					
67	TEST	IP	CLAIP1GD	No	No	100k down	-	
68	VDD	PWR	CLALLVP					
69	TIMEMARK	OP	CLAOP01L1	No	-	-	No	
70	IDDQTEST	IP	CLHIDDQ	No	No	100k down	-	
71	GND	PWR	CLALLVM					
72	RTC_XIN		CLGOSC32K					
73	RTC_XOUT		CLGOSC32K					
74	TESTMODE	IP	CLAIP1NR	No	No	None	-	
75	NSRESET	IP	CLAIP1NR	No	No	None	-	
76	U2TXD	OP	CLAOP03L1	-	-	-	No	

Pin No.	Pin Name	Pin Type	IP / OP Cell Type	5V Tol.?	Hold Cell?	Pull-up/ -down	Tri-state	Notes
77	U2RXD	IP	SCJIP1NR	Yes	No	None	-	
78	U1TXD	OP	CLAOP03L1	-	-	-	No	
79	U1RXD	IP	SCJIP1NR	Yes	No	None	-	
80	PLLGND	PWR	CLPLLVM					
81	PLLVDD	PWR	CLPLLVP					
82	GND	PWR	CLALLGND					
83	PLLAT1	OP	CLHANPOP					
84	NICE	IP	CLAIP1NR	No	No	None	-	
85	VDD	PWR	CLALLVDD					
86	TCK / bdiag[0] / XReq	IP/OP	CLAIO1NR01N	No	No	None	Yes	
87	TDI / bdiag[1] / XWrite	IP/OP	CLAIO1NR01N	No	No	None	Yes	
88	TDO / bdiag[2] /XBurst	IP/OP	CLAIO1NR01N	No	No	None	Yes	
89	TMS / bdiag[3] / XCon	IP/OP	CLAIO1NR01N	No	No	None	Yes	
90	NTRST	IP	CLAIP1NR	No	No	None	-	
91	GPIO[7] / DT1	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
92	GPIO[6]	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
93	GPIO[5]/DISCOP	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
94	GND	PWR	CLALLGND					
95	GPIO[4] /DISCIO	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
96	GPIO[3] / BSIO_SS[1]	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
97	GPIO[2] / BSIO_SS[0]	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
98	VDD	PWR	CLALLVDD					
99	GPIO[1] / BSIO_DATA	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	
100	GPIO[0] / BSIO_CLK	IP/OP	SCJIO1NR01N	Yes	No	None	Yes	

Table 20.1 GP4020 Pin Types

NOTES:-

1) Pin is set to High Impedance (Tri-state) in RF Power-down mode. RF Power-down achieved when RF_PD bit or RF_SLEEP bit in PCL POW_CNTL register set to “1”.
2) Pin set to be an Input when GP4020 in Test Mode
3) Pin set to be High Impedance (Tri-state) when GP4020 in Test Mode

20.2 Input Delays

The following tables show the delays for all Logic Inputs on the GP4020. The delays quoted below are shown at midpoint temperature (+25°C), supply voltage (+3.3V) and silicon process for all input lines in the GP4020.

Key: IP $\rightarrow$ D $\uparrow$ signifies Time from Input going High, to registered Data going High

IP $\rightarrow$ D $\downarrow$ signifies Time from Input going Low, to registered Data going Low

20.2.1 3.3V Inputs: CLAIO1HD01N, CLAIO1HD03N, CLAIO1NR01N, CLAIP1GD, CLAIP1GU, CLAIP1NR

Switching characteristics at 25°C, 3.3V supply.

Switching Delay (ns)	Input edge 0.1ns Load (fF)					Input edge 1.5ns Load (fF)				
	50	100	250	500	1000	50	100	250	500	1000
IP → D ↑	0.29	0.34	0.49	0.74	1.23	0.40	0.45	0.60	0.84	1.34
IP → D ↓	0.29	0.31	0.39	0.51	0.76	0.62	0.64	0.72	0.84	1.09

Table 20.2 3.3V Input delays

20.2.2 5V Tolerant Inputs: SCJIO1NR01N, SCJIP1NR

Switching characteristics at 25°C, 3.3V supply.

Switching Delay (ns)	Input edge 0.1ns Load (fF)					Input edge 1.5ns Load (fF)				
	50	100	200	400	800	50	100	200	400	800
IP → D ↑	0.31	0.36	0.47	0.68	1.11	0.38	0.44	0.54	0.76	1.18
IP → D ↓	0.58	0.60	0.65	0.74	0.93	0.85	0.87	0.92	1.01	1.20

Table 20.3 5V Tolerant Input delays

20.3 Output Delays

All delays quoted below are shown at midpoint temperature (+25°C), supply voltage (+3.3V) and silicon process for all logic / data output lines in the GP4020.

Key:
- D → OP ↑ signifies Time from Data in going High, to Output going High
- D → OP ↓ signifies Time from Data in going Low, to Output going Low
- T → OP ↑ signifies Time from Tri-state Hi Impedance disable, to Output going High
- T → OP ↓ signifies Time from Tri-state Hi Impedance disable, to Output going Low

20.3.1 X01 Drive Outputs (x1 Current drive)

20.3.1.1 Slow L1 outputs: CLAOP01L1.

Switching Delay (ns)	Input edge 0.1ns Load (pF)					Input edge 1.5ns Load (pF)				
	10	20	40	80	150	10	20	40	80	150
D → OP ↑	6.03	7.72	10.36	14.03	18.74	5.75	7.45	10.09	13.76	18.47
D → OP ↓	8.09	9.52	11.73	14.75	18.56	8.90	10.33	12.54	15.57	19.38
T → OP ↑	6.32	8.02	10.67	14.38	19.16	7.13	8.83	11.49	15.19	19.97
T → OP ↓	8.09	9.51	11.71	14.72	18.52	8.89	10.32	12.52	15.53	19.33

Table 20.4 X01 Slow L1 3.3V Output delays

20.3.1.2 Normal N outputs (3.3V outputs): CLAIO1HD01N, CLAIO1NR01N, CLAOP01N.

Switching Delay (ns)	Input edge 0.1ns Load (pF)					Input edge 1.5ns Load (pF)				
	10	20	40	80	150	10	20	40	80	150
D → OP ↑	5.82	6.99	8.89	11.75	15.76	5.55	6.72	8.62	11.48	15.49
D → OP ↓	5.55	6.75	8.60	11.18	14.48	6.37	7.56	9.42	11.99	15.30
T → OP ↑	6.10	7.27	9.18	12.05	16.06	6.91	8.08	9.99	12.86	16.87
T → OP ↓	5.49	6.66	8.49	11.04	14.33	6.30	7.47	9.30	11.85	15.15

Table 20.5 X01 Normal N 3.3V Output delays

20.3.1.3 Normal N outputs (5V Tolerant outputs): SCJIO1NR01N.

Switching Delay (ns)	Input edge 0.1ns Load (pF)					Input edge 1.5ns Load (pF)				
	10	20	40	80	150	10	20	40	80	150
D → OP ↑	6.22	6.99	8.22	10.00	12.38	5.88	6.65	7.88	9.66	12.04
D → OP ↓	8.49	9.42	10.82	12.66	14.86	9.35	10.28	11.68	13.52	15.72
T → OP ↑	6.40	7.17	8.40	10.17	12.56	7.26	8.03	9.26	11.04	13.42
T → OP ↓	7.28	8.19	9.57	11.38	13.57	8.14	9.04	10.42	12.24	14.43

Table 20.6 X01 Normal N 5V Tolerant Output delays

20.3.2 X03 Drive Outputs (x3 Current drive)

20.3.2.1 Slow L1 outputs: CLAOP03L1.

Switching Delay (ns)	Input edge 0.1ns Load (pF)					Input edge 1.5ns Load (pF)				
	10	20	40	80	150	10	20	40	80	150
D → OP ↑	6.03	7.72	10.36	14.03	18.74	5.75	7.45	10.09	13.76	18.47
D → OP ↓	8.09	9.52	11.73	14.75	18.56	8.90	10.33	12.54	15.57	19.38
T → OP ↑	6.32	8.02	10.67	14.38	19.16	7.13	8.83	11.49	15.19	19.97
T → OP ↓	8.09	9.51	11.71	14.72	18.52	8.89	10.32	12.52	15.53	19.33

Table 20.7 X03 Slow L1 3.3V Output delays

20.3.2.2 Normal N outputs: CLAIO1HD03N, CLAOP03N.

Switching Delay (ns)	Input edge 0.1ns Load (pF)					Input edge 1.5ns Load (pF)				
	10	20	40	80	150	10	20	40	80	150
D → OP ↑	5.82	6.99	8.89	11.75	15.76	5.55	6.72	8.62	11.48	15.49
D → OP ↓	5.55	6.75	8.60	11.18	14.48	6.37	7.56	9.42	11.99	15.30
T → OP ↑	6.10	7.27	9.18	12.05	16.06	6.91	8.08	9.99	12.86	16.87
T → OP ↓	5.49	6.66	8.49	11.04	14.33	6.30	7.47	9.30	11.85	15.15

Table 20.8 X03 Normal N 3.3V Output delays

20.4 Cell DC Characteristics

The characteristics in *Table 20.9 below* apply to the generic logic input and output cells, of types:

a) Input - 3.3V: CLAIO1HD01N, CLAIO1HD03N, CLAIO1NR01N, CLAIP1GD, CLAIP1GU, CLAIP1NR

b) Input – 5V: SCJIO1NR01N, SCJIP1NR,

c) Output – X01 – Slow: CLAOP01L1,

d) Output – X01 – Normal: CLAIO1HD01N, CLAIO1NR01N, CLAOP01N, SCJIO1NR01N

e) Output – X03 – Slow: CLAOP03L1,

f) Output – X03 – Normal: CLAIO1HD03N, CLAOP03N

Typical characteristics are with Vdd = +3.3v, Temp = +27°C and typical silicon process. The min. and max. characteristics are defined over ALL silicon process conditions, Vdd = +3.0V to +3.6V and Temp = -40°C to +85°C.

Parameter	Cell Type	Min	Typ	Max	Unit	Conditions
Input Leakage	All IP	-1		+1	μA	No Pull Up/Down, VDD = 3.6V
Output (Tristate) Leakage	All OP			1	μA	No Pull Up/Down, VDD = 3.6V
Input Capacitance	All IP		5		pF	Not including Package
Output Capacitance	All OP		5		pF	Not including Package
Weak Pull-up Current	GU IP		-30		μA	Input at 0V
Weak Pull-down Current	GD IP		30		μA	Input at VDD
Input Hold-Up Current	HD IP		-24		μA	Vin = 55% of VDD
Input Hold-Down Current	HD IP		19		μA	Vin = 30% of VDD
Input Hold Threshold	HD IP		1.48		V	VDD = 3.3V
Low Input Level – CMOS VIL	All IP			0.3 Vdd	V	3.0<VDD<3.6
High Input Level – CMOS VIH	All IP	0.7 Vdd			V	3.0<VDD<3.6
Low Output Level – CMOS VOL	All OP			0.4	V	3.0<VDD<3.6
High Output Level – CMOS VOH	All OP	2.4			V	3.0<VDD<3.6
Low Output Current IOL	X01 OP		2		mA	
High Output Current IOH	X01 OP		2		mA	
Low Output Current IOL	X03 OP		6		mA	
High Output Current IOH	X03 OP		6		mA	

Table 20.9 Input & Output Cell DC Characteristics

This Page intentionally left blank.

21 TIMING CHARACTERISTICS

The timing parameters in this section assume a logic switching point of 50% of VDD:

All inputs assume rise and fall times of nominally 2ns.

Minimum (min) and maximum (max) figures are referenced at extremes of Voltage and Temperature.

Important Notes:

1) All parameters will scale from their MIN to MAX value as temperature RISES or voltage FALLS. Their relationship, however, will always remain the same. Therefore, if conditions give rise to a maximum propagation delay, such as Taddr (max), any corresponding hold times will also be maximum e.g. Tnweh (max).

2) All timings in the following section are preliminary figures based on simulation results under worst/best case conditions where appropriate.

3) Min and Max delays depend upon temperature range, supply voltage, input edges speed and process spreads. Accurate delays are calculated by Zarlink design kits on approved simulators.

21.1 Memory Peripheral Controller (MPC) External Read & Write timing parameters with on-chip Wait-state Control

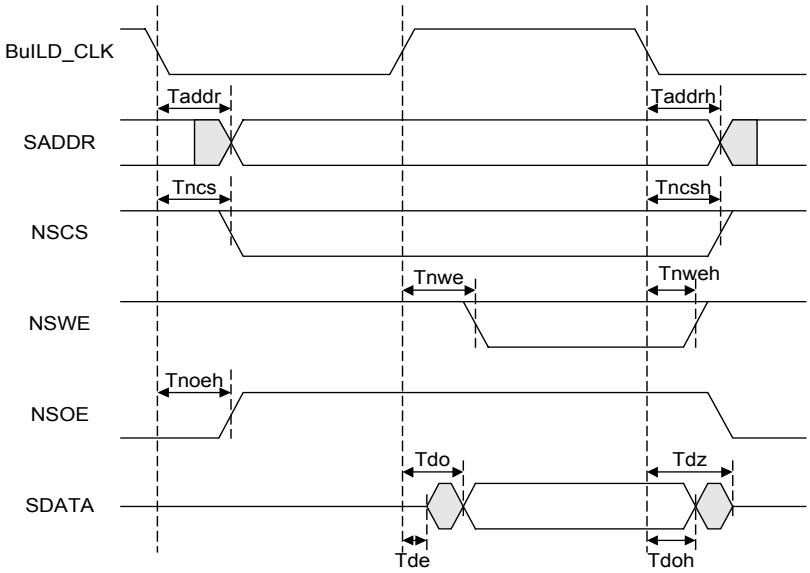


Figure 21.1 MPC Timing Diagram - External Memory Write Cycle

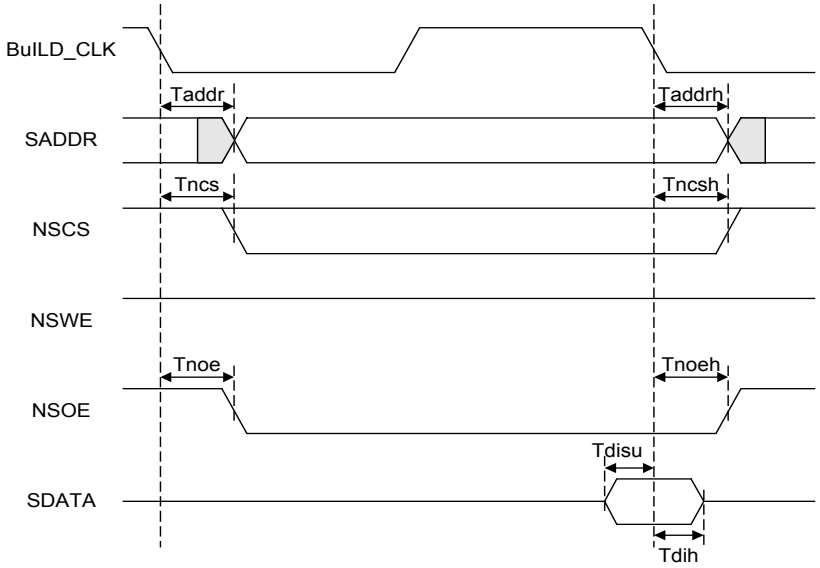


Figure 21.2 MPC Timing Diagram - External Memory Read Cycle

Note: These MPC Write and Read transactions are the same whether the ARM7TDMI core or the DMA Controller is the current bus master.

Parameter	Min	Max	Unit	Description and notes
Tsclk	14.0		ns	BμILD Clk low period
Tsclkh	14.0		ns	BμILD Clk high period
Taddr	12.2	31.5	ns	BμILD Clk to Address Valid
Taddrh	10.7	28.0	ns	Address hold after Sclk
Tncs	11.6	30.0	ns	BμILD Clk to chip select valid
Tncsh	10.5	27.6	ns	Chip select hold after Sclk
Tnoe	13.1	33.0	ns	BμILD Clk to output enable active
Tnoeh	10.7	27.7	ns	Output enable hold after Sclk
Tnwe	9.4	24.4	ns	BμILD Clk to Write enable
Tnweh	9.9	26.0	ns	Write enable hold after Sclk
Tdisu	-2.0	-	ns	Data setup before Sclk
Tdih	7.5	-	ns	Data input hold time
Tde	9.5	30	ns	Data enable time after Sclk
Tdo	12.8	31.6	ns	Data valid time after Sclk
Tdz	11.4	29.3	ns	Data disable time after Sclk
Tdoh	10.8	28.4	ns	Data out hold time after Sclk

Table 21.1 Simulated Timing parameters for MPC External Transactions with on-chip Wait-state control

Notes:

1) MIN results simulated for -40°C, fast silicon process, Vdd = +3.6V, output loading = 50pF (except Tdisu and Tdih).

2) MAX results simulated for +85°C, slow silicon process, Vdd = +3.0 V, output loading = 50pF.

3) Tnweh (NSWE rising) will ALWAYS precede Taddrh and Tncsh.

21.2 Memory Peripheral Controller (MPC) External Read & Write timing parameters with SWait Control

Memory accesses with SWait control have default of one wait-state access, in addition to additional wait-states triggered by an external SWait = High signal.

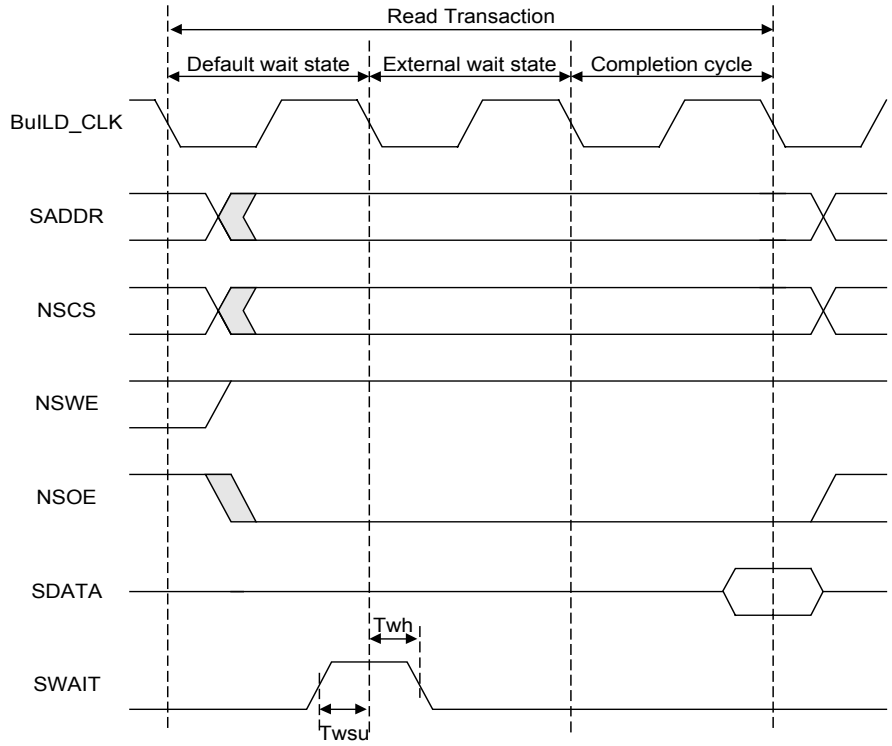


Figure 21.3 MPC Timing Diagram - External Memory Read Cycle with external Wait-State control

Parameter	Min	Max	Unit	Description and notes
Twsu	0		ns	SWait setup time before BμILD CLK
Twh	7.5		ns	SWait hold time after BμILD CLK

Table 21.2 Simulated SWait Timing parameters for MPC External Transactions

21.3 Direct Memory Access Controller (DMAC) single address transfer timing

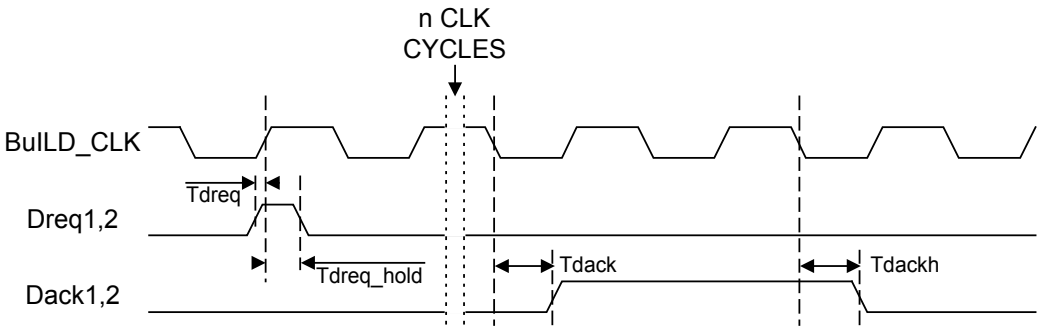


Figure 21.4 DMAC timing: Single address transfer.

For this example an edge triggered packet transfer (size = 2) is shown.

NOTE: When performing a DMA transfer, memory signals are as per the MPC timing information.

Parameter	Min	Max	Unit	Description and notes
Tdreq	0		ns	Dreq setup before BμILD_CLK
Tdreq_hold	7.5		ns	Dreq hold time after BμILD_CLK
Tdack	14.2	29.8	ns	BμILD_CLK to Dack active
Tdackh	11.1	29.0	ns	Dack hold after BμILD_CLK (Dack1 & Dack2)

Table 21.3 Simulated DMAC Timing parameters

Notes:

1) MIN results simulated for -40°C, fast silicon process, Vdd = +3.6V, output loading = 50pF, used for input hold.

2) MAX results simulated for +85°C, slow silicon process, Vdd = +3.0 V, output loading = 50pF, used for input setup.

21.4 External interrupt inputs: Timing for Edge sensitivity mode

Interrupts are asynchronous and therefore unaffected by BμILD_CLK.

Interrupts are programmable and timings apply for both rising and falling edges.

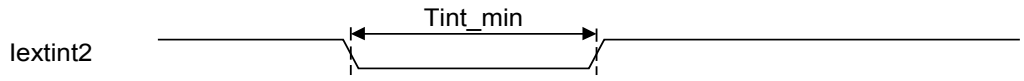


Figure 21.5 Interrupt timing

Parameter	Min	Max	Unit	Description and notes
Tint_min	5.0	-	ns	Minimum external interrupt width

Table 21.4 Simulated Interrupt Timing parameters

Note: MIN result simulated for -40°C, fast silicon process, Vdd = +3.6V, output loading = 50pF, used for input hold.

21.5 External interrupt inputs: Timing for Level sensitivity mode

When Level-sensitive interrupts are programmed, they must remain active until a suitable response is generated. (i.e. until they have been serviced.) Therefore, please refer to “Firefly MF1 Core Design Manual” for timing relationship diagrams.

21.6 System Services Module (SSM) Broadcast Diagnostic Timing Diagrams

The SBDIAG lines referred to here are the Xdiag[3:0] lines which can be configured within the SSM to be multiplexed with the JTAG interface, to allow access to any SADD or SDATA line within the Firefly MF1.

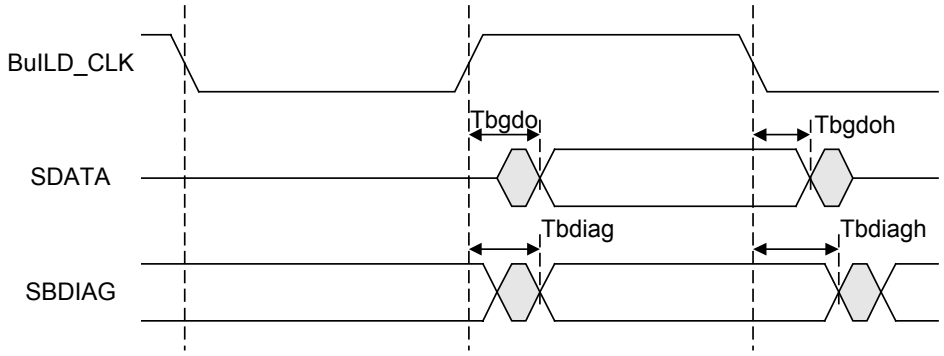


Figure 21.6 External Broadcast diagnostic signal (SBDIAG) timings from SSM.

Parameter	Typ.	units	Description and notes
Tbgdo	10.0	ns	Sdata valid after BμILD_CLK with broadcast diagnostics enabled
Tbgdoh	8.0	ns	Sdata hold time after BμILD_CLK with broadcast diagnostics enabled
Tbdiag	15.0	ns	Bdiag data valid after BμILD_CLK
Tbdiagh	13.0	ns	Bdiag output data hold time after BμILD_CLK

Table 21.5 Simulated Broadcast Diagnostic Timing parameters

Note: Typical results simulated for +25°C, typical silicon process, Vdd = +3.3 V, output loading = 50pF.

21.7 JTAG interface Timing Diagram

The data for the JTAG interface timing has been copied from Rev 3 of the ARM7TDMI Technical Reference Manual (document reference ARM DDI 0029F), which is downloadable (1.7 MB PDF) from ARM's website http://www.arm.com. The documentation download page can be found at: http://www.arm.com/arm/documentation?OpenDocument .

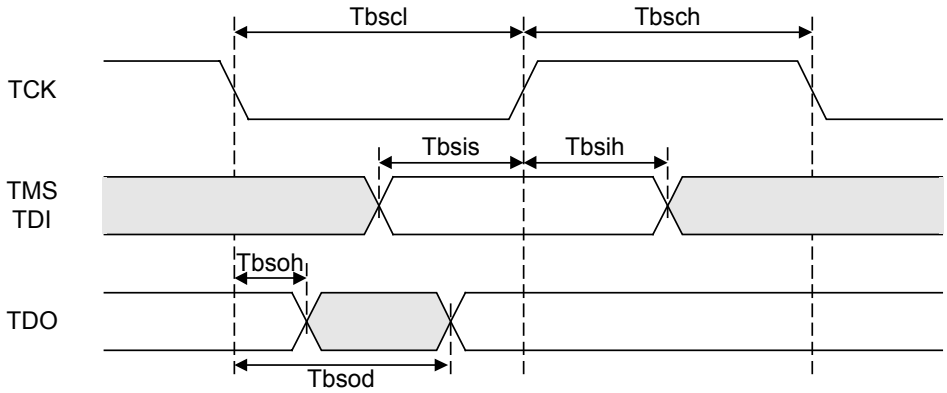


Figure 21.7 JTAG Interface Characteristics

Parameter	Min	Max	units	Description and notes
Tbscl	15.6	-	ns	TCK low period
Tbsch	15.6	-	ns	TCK high period
Tbsis	5.0	-	ns	TDI,TMS setup to [TCr]
Tbsih	5.0	-	ns	TDI,TMS hold from [TCr]
Tbsoh	2.4	-	ns	TDO hold time
Tbsod	-	25	ns	TCr to TDO valid
Tbsr	25	-	ns	Reset period

Table 21.6 JTAG Timing parameters

Notes:

1) For correct data latching, the I/O signals (from the core and the pads) must be set-up and held with respect to the rising edge of TCK in the CAPTURE-DR state of the INTEST and EXTEST instructions.

2) Assumes that the data outputs are loaded with the AC test loads (see AC parameter specification).

INDEXES

This page intentionally left blank

Table of Figures

Page

This page intentionally left blank

Table of Data Tables

Page

This page intentionally left blank

For more information about all Zarlink products
visit our Web Site at

www.zarlink.com

TECHNICAL DOCUMENTATION - NOT FOR RESALE