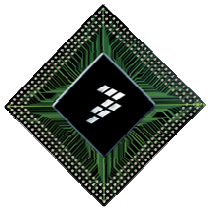
TWR-MCF52259-Ethenet



Hareesh S
Sr.FAE

freescale™
semiconductor

Ethernet Overview Session

Ethernet Router

Ethernet Switch

Ethernet NIC

Ethernet Cables

Ethernet Connector

freescale™ semiconductor

Ethernet Overview Session

What is Ethernet?

- It's a cable I connect to my computer to surf the net
- It's how I do emails
- My home router uses it to let all my computers talk

Why do we care about Ethernet?

- Work is telling me I need it for my embedded product
- It will let me remotely access my embedded product
- Seems to be a cool way to have fast downloads

How will I use Ethernet?

- Just in factory application (i.e. local only)
- Connected to WLAN (i.e. publicly accessible)
- Through VPN only (i.e. secure tunnel)

freescale™ semiconductor

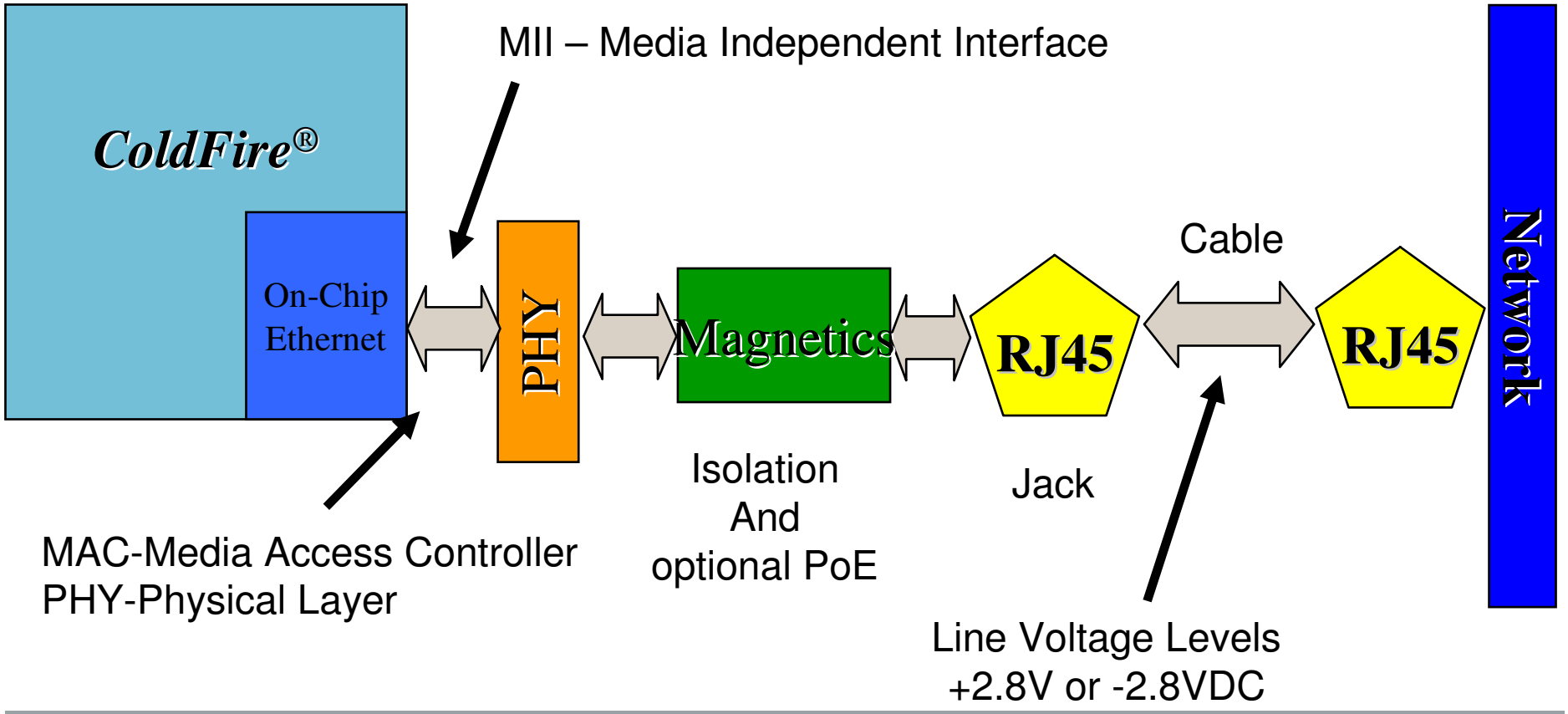
Ethernet

Ethernet defines the mechanical/electrical connection between devices (the physical layer).

Ethernet also defines a protocol used to communicate between multiple devices (the MAC layer).

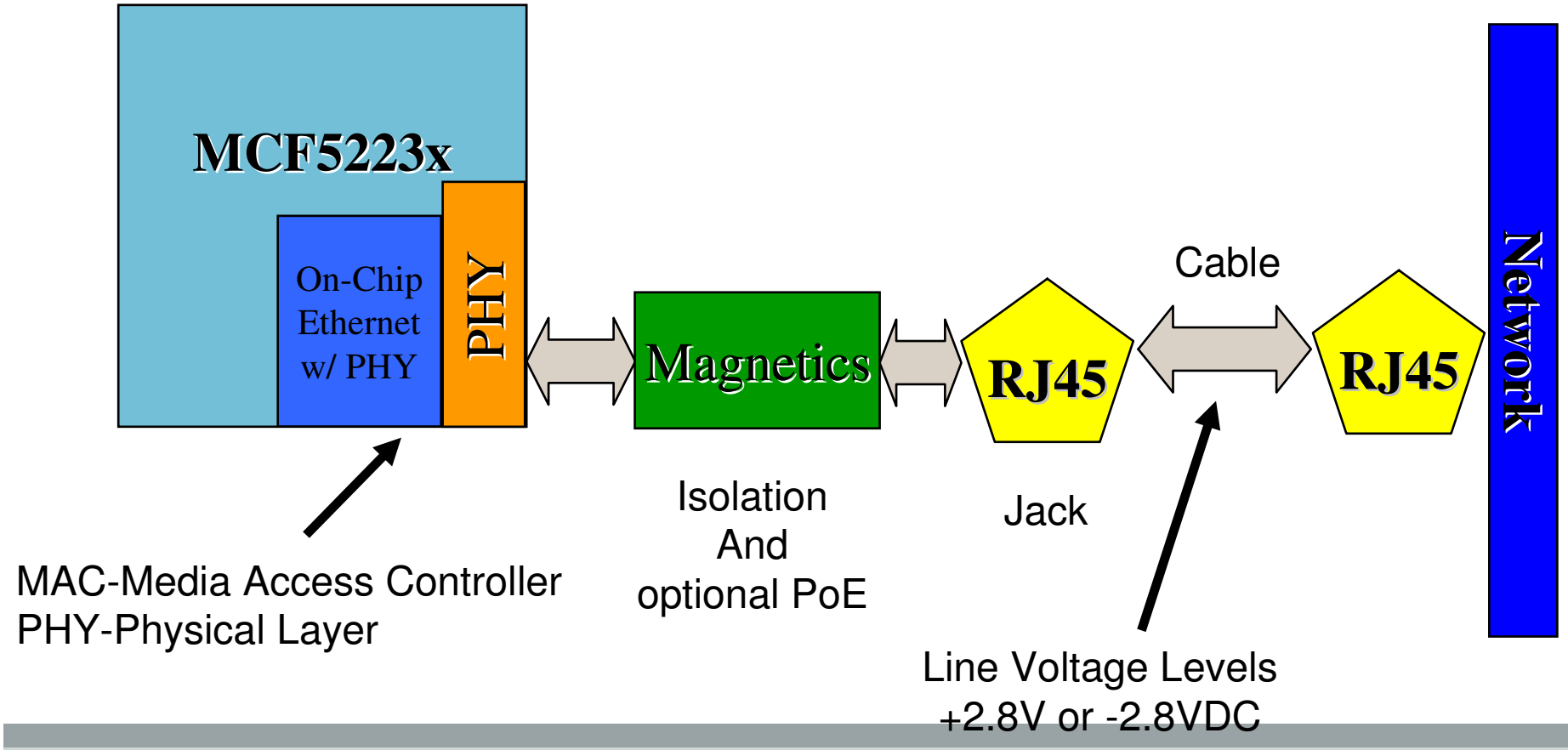
Ethernet is defined by the IEEE 802.3 standard

Generic *ColdFire®* Board Layout of Ethernet



freescale™ semiconductor

M52233DEMO Board Layout of Ethernet

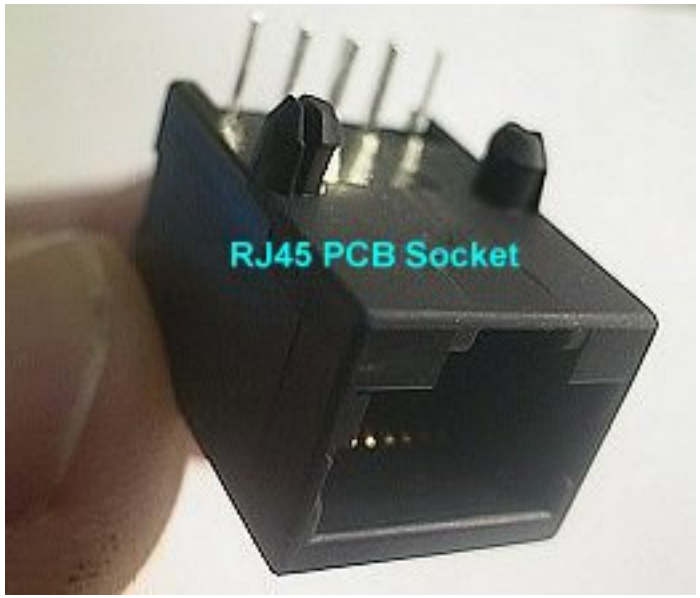


freescale™ semiconductor

Ethernet Overview Physical Session

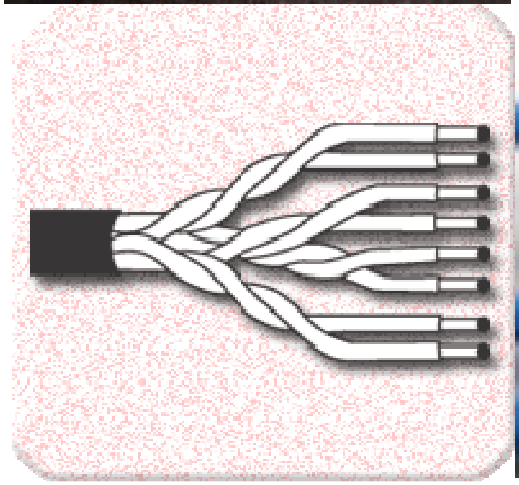
Connectors

- RJ-45



Cables

- CAT-5
  - 24 AWG solid bare copper
  - Four unbounded twisted pairs

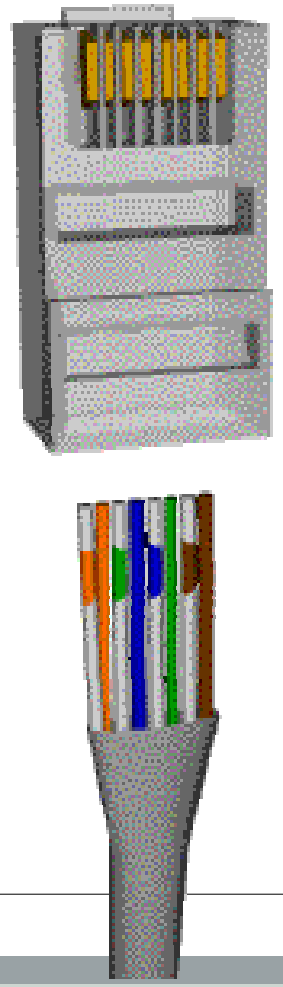


freescale semiconductor

Ethernet Cable: Straight Through Pinout

The following table demonstrates the proper color scheme.

Wire pair #1:	**White/Blue Blue**
Wire pair #2:	**White/Orange Orange**
Wire pair #3:	**White/Green Green**
Wire pair #4:	**White/Brown Brown**



RJ-45 Pin	Signal	Direction	RJ-45 Pin
1	TX+	--->	1
2	TX-	--->	2
3	RX+	<---	3
4	-	-	4
5	-	-	5
6	RX-	<---	6
7	-	-	7
8	-	-	8

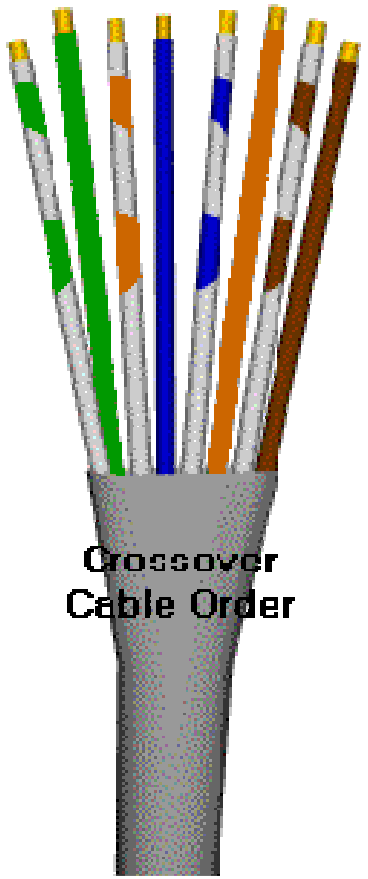
Source: http://www.netspec.com/helpdesk/wiredoc.html

freescale™ semiconductor

Ethernet Cable: Crossover Pinout

The following is the proper pin out and cable pair/color order for the "crossover" end.

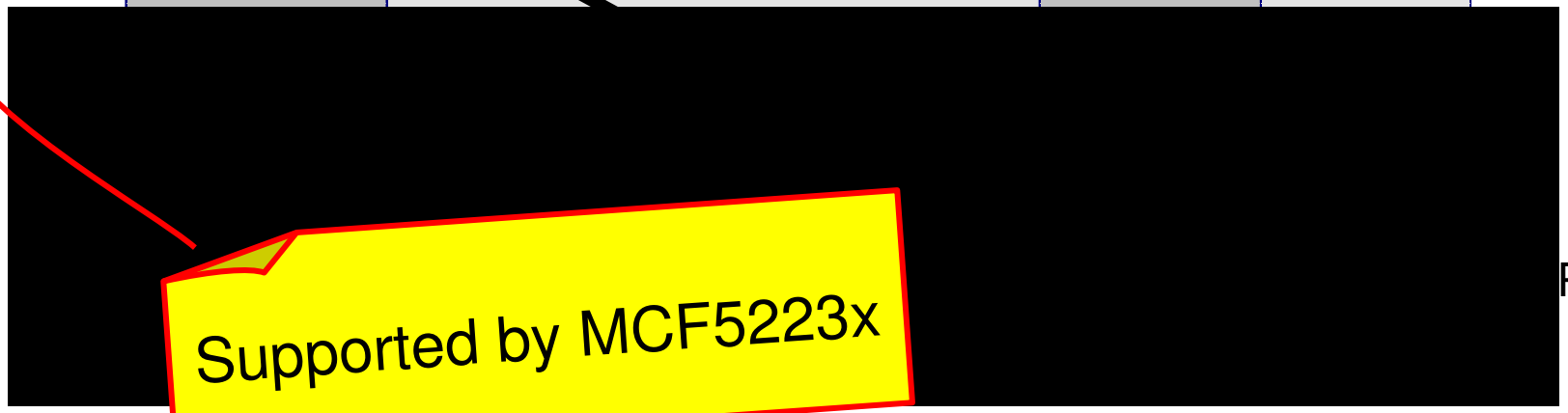
Pair#2 is connected to pins 1 and 2 like this:	
Pin 1 wire color:	**white/green**
Pin 2 wire color:	**green**
Pair#3 is connected to pins 3 and 6 like this:	
Pin 3 wire color:	**white/orange**
Pin 6 wire color:	**orange**



freescale™ semiconductor

* Distance Signal Travels at 100 Mbits?

Name	Transmission Medium	Data Rate (Mb/s)	Distance (m)
100BASE-TX	2 pairs of Category UTP-5, alternative 2 pairs of STP, 150 Ω Impedance, Cable Code MLT-3, Full Duplex	100	100
100BASE-FX	2 Multimode Optical Fiber (62.5/125 µm), Cable Code 4B5B, NRZI, Full Duplex	100	2,000



Supported by MCF5223x

PHY

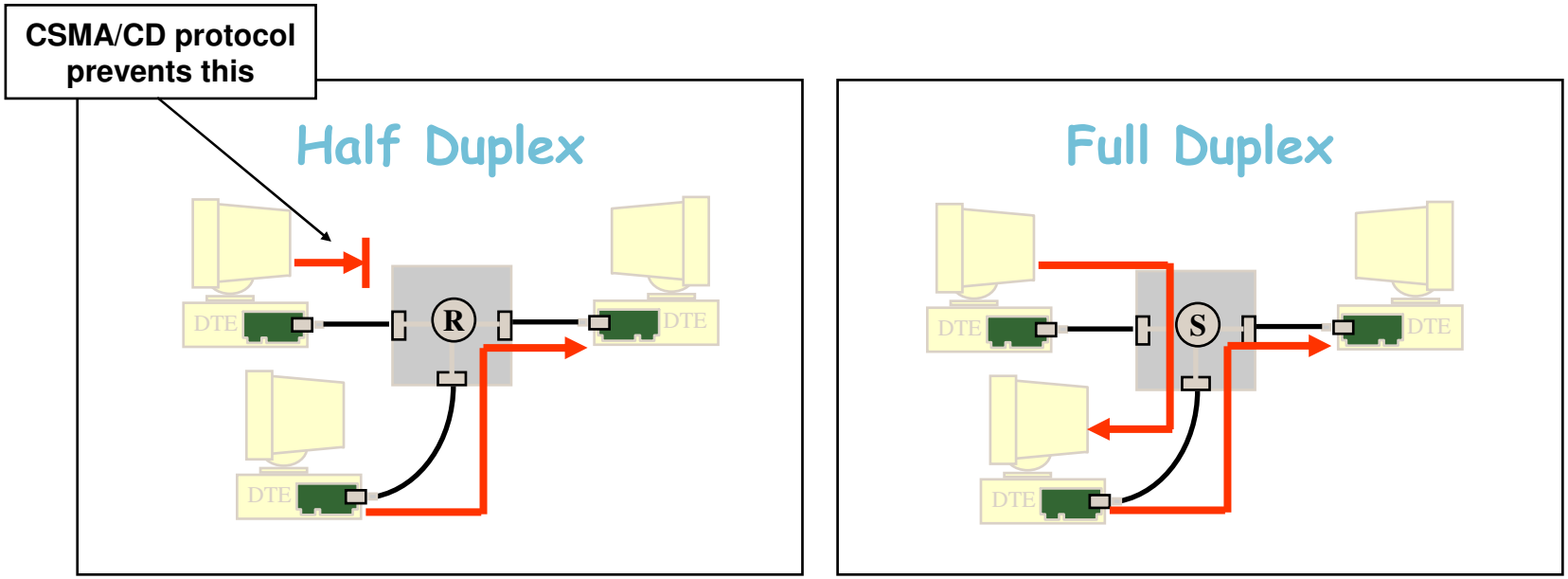
freescale™ semiconductor

Full Duplex Ethernet Links

Full duplex operation means that devices at each end of a full duplex link can send and receive data simultaneously.

This means, theoretically, that Full Duplex has twice the bandwidth of normal (half duplex) Ethernet.

Since there are only two devices on a full duplex link, there is no shared channel and no collisions.



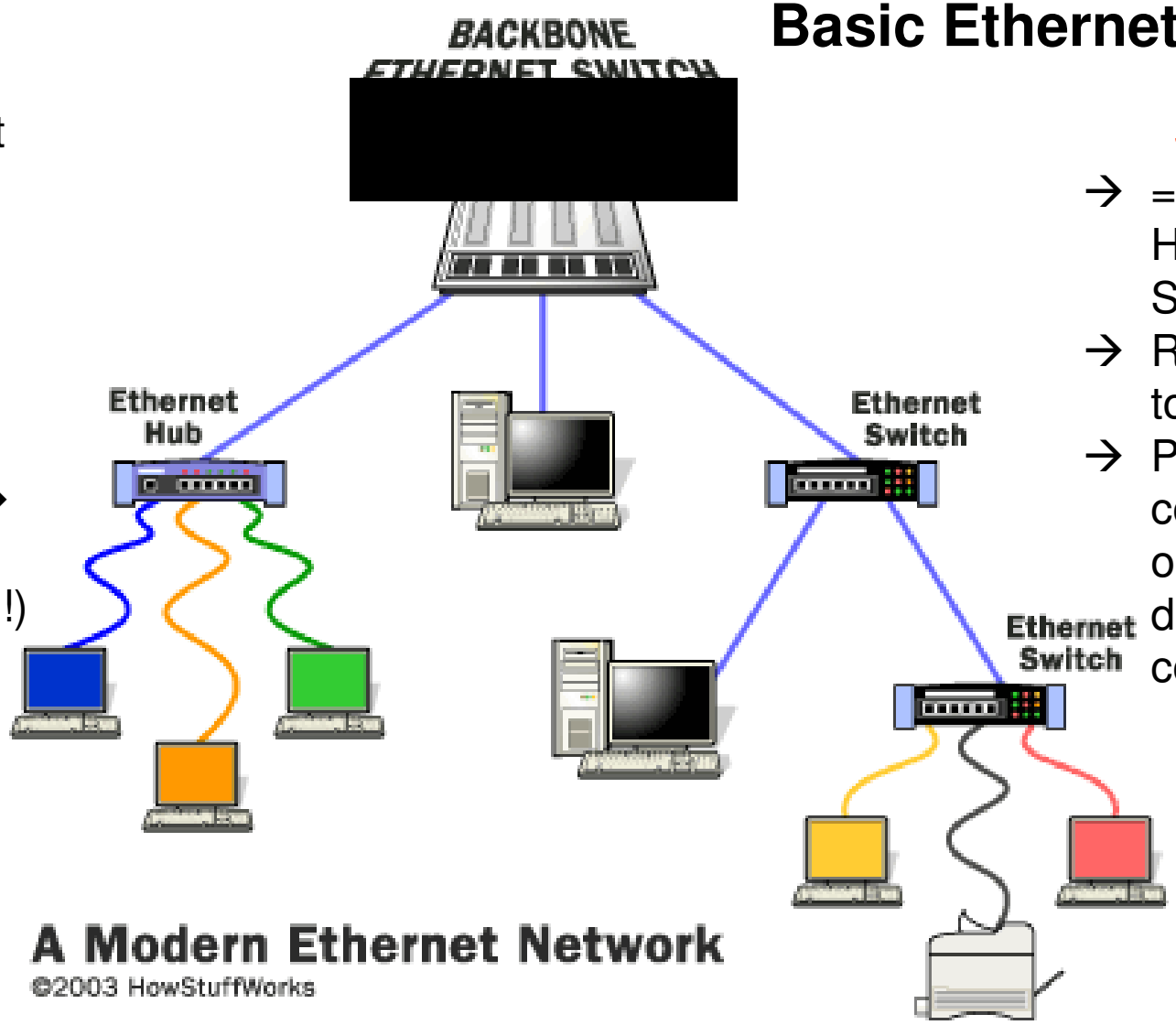
freescale semiconductor

Basic Ethernet Network

Hub

- → = Multiport repeater
- → physically star topology (common transport medium → CSMA/CD!)
- → Logically bus topology

Outputs all incoming signals on all outputs

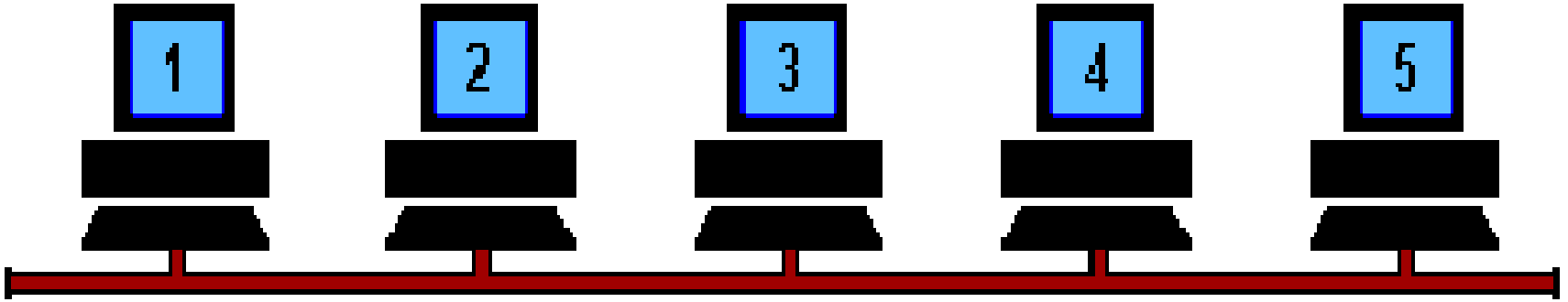


Switch

- → = Switching Hub, LAN Switch
- → Real star topology
- → Physical connection only for the duration of the communication

freescale semiconductor

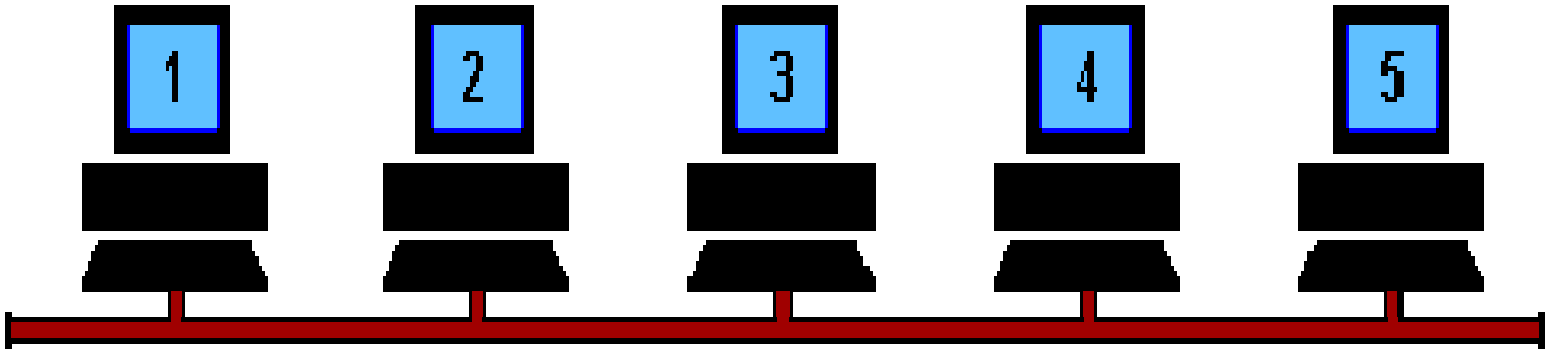
Basic Ethernet Bus



Co-axial based Ethernet connection daisy chain connection

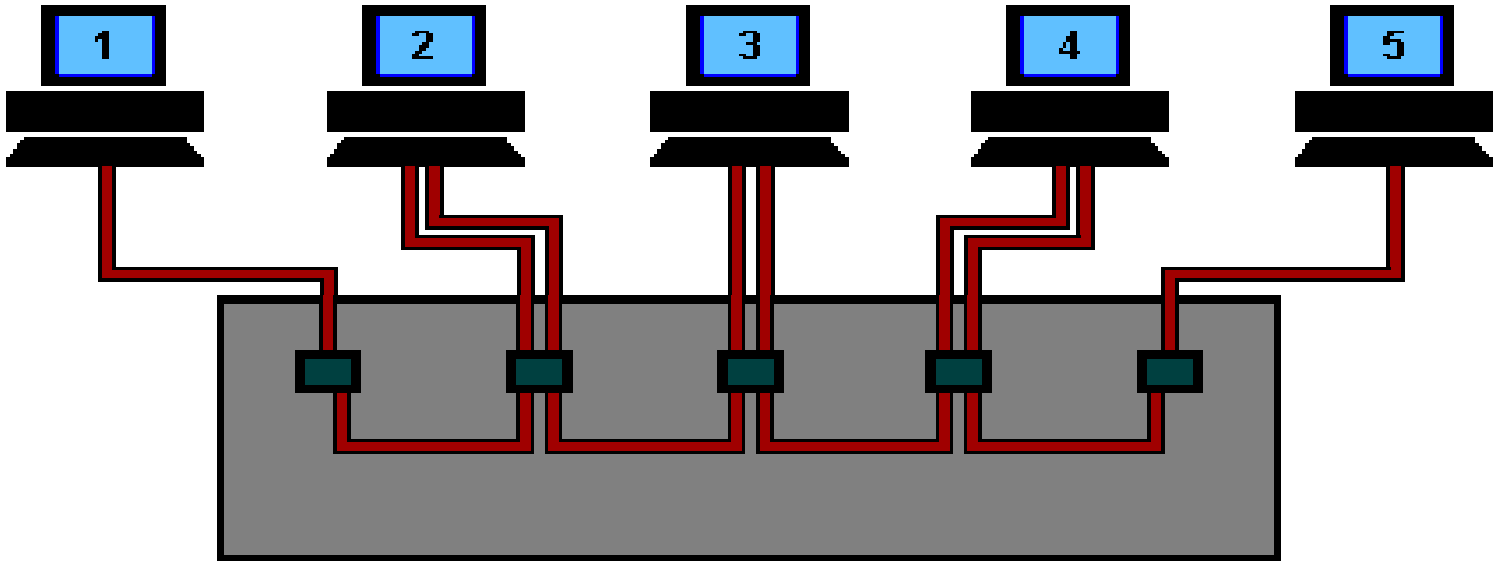
freescale™ semiconductor

Collisions



freescale™ semiconductor

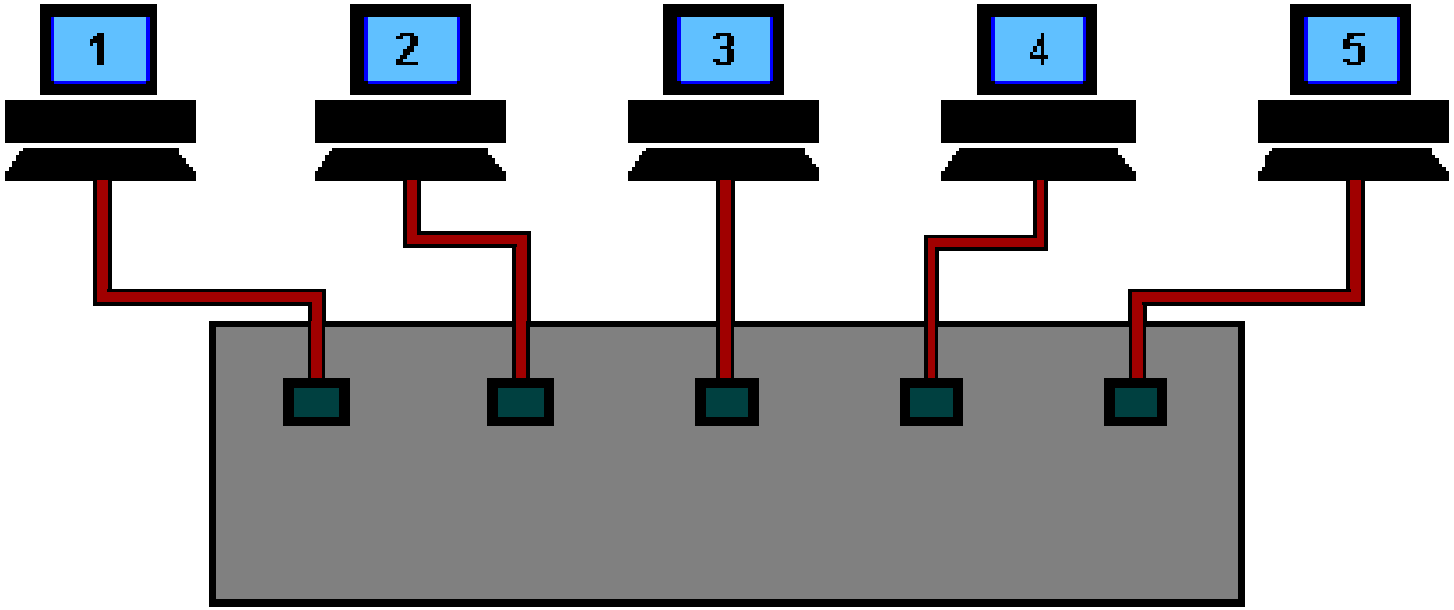
HUB



Centralized connection
Can bypass not connected

freescale™ semiconductor

Switch



The switch reads the destination addresses and 'switches' the signals directly to the recipients without broadcasting to all of the machines on the network.
This 'point to point' switching alleviates the problems associated with collisions and considerably improves network speed.

freescale™ semiconductor

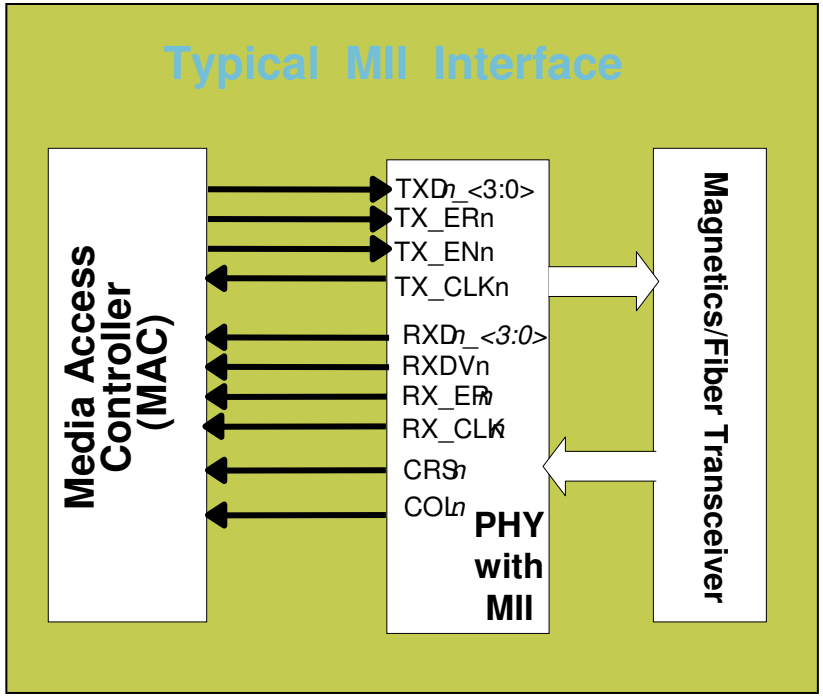
Ethernet Router/Gateway

- A Router or Gateway is used to translate one protocol to another.
- It is also used when the physical layer changes mediums.
  - Ethernet to fiber
- At one time there was a difference between a router and a gateway.
  - The gateway was strictly used as a medium translator (electrical) and the router was strictly used as protocol translator (software).
  - Now routers and gateways are normally combined and called a routers.

Note: Ethernet to WiFi is a router functionality.

*Media Independent Interface (MII)

- The MII links the Ethernet MAC with the PHY.
- An MII may support both 10-Mb/s and 100-Mb/s operation, allowing network devices to connect to both 10BASE-T and 100BASE-T media segments.
- The MII electronics may be linked to an outboard transceiver through a 40-pin MII connector and a short (0.5m) MII cable.
- The MII is internally connected to the EPHY on the MCF5223x



- 4-bit wide Tx and Rx data @2.5MHz or 25MHz
- TTL signal levels
- Full duplex
- Media status signals:
- Carrier Presence & Collision

freescale™ semiconductor

*Autonegotiation

Auto-Negotiation is the exchange of information about each stations abilities over a link segment allows the stations to achieve the best possible mode of operation.

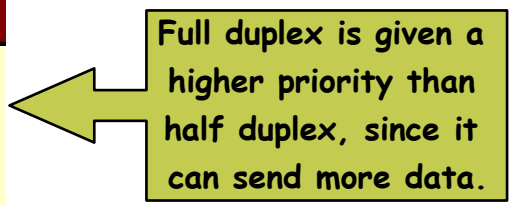
The highest performance mode of operation that Auto-Negotiation can achieve is based on a priority table.

The Auto-Negotiation protocol contains a set of priorities which result in the devices selecting their highest common set of abilities.

If the devices at both ends of the link can support full duplex operation, and if they also both support Auto-Negotiation of this capability, then they will automatically configure for full duplex.

The priorities are listed in the table below...

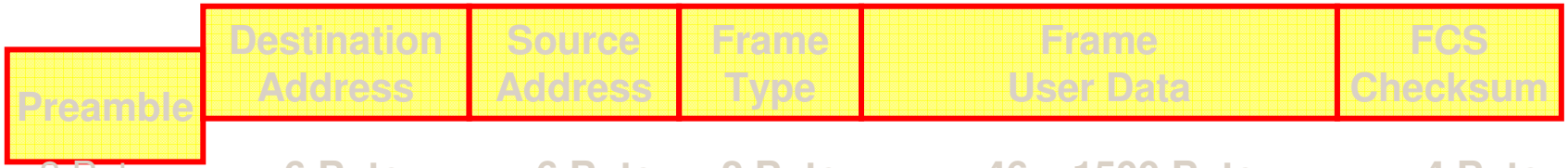
Auto-Negotiation priority Resolution Table	
A	100 Base-TX Full Duplex
B	100 Base-T4
C	100 Base-TX
D	10 Base-T Full Duplex
E	10 Base-T Half Duplex



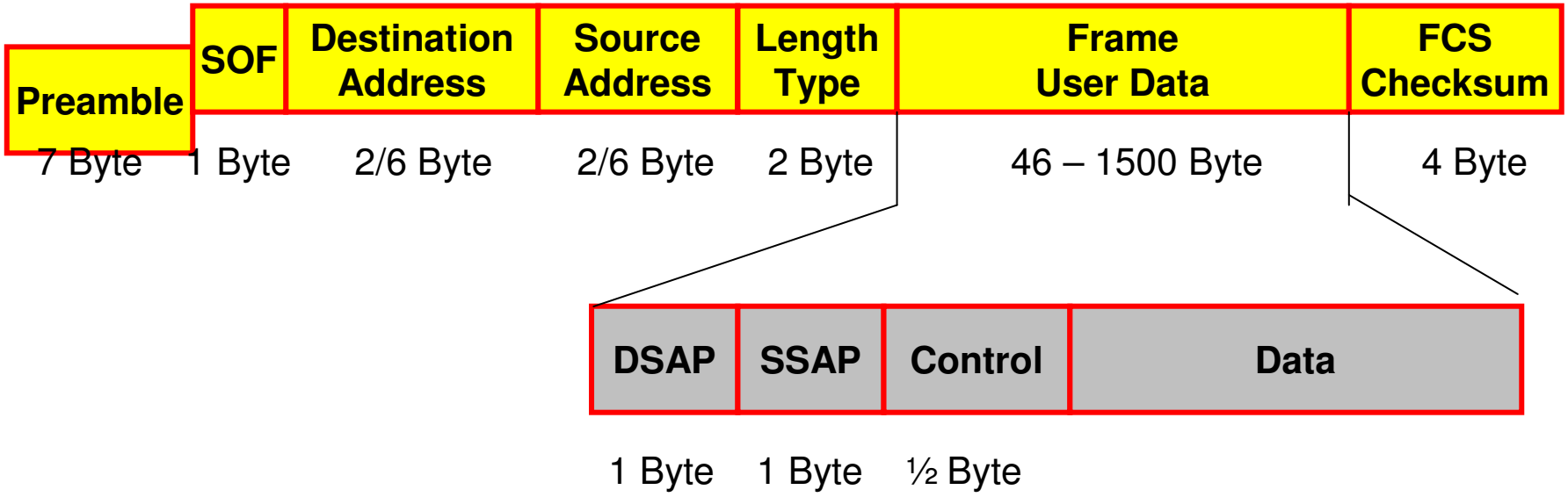
freescale™ semiconductor

The Ethernet Data Packet Format

Ethernet Data Frame – old/original format used



IEEE 802.3 Data Frame



freescale semiconductor

Terminology

CSMA - Carrier Sense Multiple Access
CD - Collision Detection
OSI - Open Systems Interconnection
ISO - International Organization for Standardization
LAN - Local Area Network
WAN- Wide Area Network
MAC - Medium Access Control
BD - Buffer Descriptor
PHY - Physical Layer Device
MDI - Medium Dependent Interface

CRC - Cyclic Redundancy Checking
FCS - Frame Checksum
IP - Internet Protocol
TCP - Transmission Control Protocol
UDP - User Datagram Protocol
ICMP - Internet Control Message Protocol
FEC - Fast Ethernet Controller
MII - Media Independent Interface
AUI - Attachment Unit Interface
DTE - Data Terminal Equipment
MAU - Medium Attachment Unit

Ethernet Definition…

Ethernet - http://dictionary.reference.com/search?q=Ethernet

<networking> A local area network first described by Metcalfe & Boggs of Xerox PARC in 1976. Specified by DEC, Intel and XEROX (DIX) as IEEE 802.3 and now recognised as the industry standard.

Data is broken into packets and each one is transmitted using the CSMA/CD algorithm until it arrives at the destination without colliding with any other packet. The first contention slot after a transmission is reserved for an acknowledge packet. A node is either transmitting or receiving at any instant. The bandwidth is about 10 Mbit/s. Disk-Ethernet-Disk transfer rate with TCP/IP is typically 30 kilobyte per second.

Version 2 specifies that collision detect of the transceiver must be activated during the inter-packet gap and that when transmission finishes, the differential transmit lines are driven to 0V (half step). It also specifies some network management functions such as reporting collisions, retries and deferrals.

Ethernet cables are classified as "XbaseY", e.g. 10base5, where X is the data rate in Mbps, "base" means "baseband" (as opposed to radio frequency) and Y is the category of cabling. The original cable was 10base5 ("full spec"), others are 10base2 ("thinnet") and 10baseT ("twisted pair") which is now (1998) very common. 100baseT ("Fast Ethernet") is also increasingly common

freescale semiconductor

More Ethernet References

Web sites:

- http://www.tcpipguide.com/
- http://www.uni-trier.de/infos/ether/ethernet-guide/ethernet-guide.html#HDR%202.0%20%20%202%2062
- http://www.lauraknapp.com/presentation.htm
- http://www.ethermanage.com/ethernet/ethernet.html
- http://osiris.sunderland.ac.uk/online/ethernet/ethernet.html
- http://computer.howstuffworks.com/ethernet.htm

References:

Ethernet, The Definitive Guide
Charles E. Spurgeon
O'Reilly
2000
ISBN 1-56592-660-9

freescale semiconductor

IP - Internet Protocol

The IP defines how a network of more then 2 devices is formed. IP is the network Layer.

IPv4 uses 32 bit addressing

IPv6 uses 128 bit addressing

A IPv4 node is defined by its IP address, and subnet mask.

- IPv4 sample address 192.168.1.0 subnet 255.255.255.0

A IPv6 node is defined by its IP address

- 2001:0DB8:0000:0000:0000:0000:1428:57ab

IP Classes

With IP V4, there are not enough IP addresses for everbody.
To solve this problem, subnetting is used.
IP addresses consists of 2 parts, a node address and a network address.
The class of the address and the subnet mask determine which part belongs to the network address and which part belongs to the node address.
Each class is defined by the first 4 bits of the IP address.

- Class A = 0xxx, or 1 to 126
- Class B = 10xx, or 128 to 191
- Class C = 110x, or 192 to 223
- Class D = 1110, or 224 to 239
- Class E = 1111, or 240 to 254

IP Subnetting

Each IP address contains a node address and a network address.
The subnet mask determines which bits identify a node address, and which bits identify a network address.
The network bits are the 1's, the node bits are the 0's.
Default subnet masks

- Class A – 255.0.0.0 — 255 networks, > 16million nodes
- Class B – 255.255.0.0 — 64K networks, 64K nodes
- Class C – 255.255.255.0 — >16 million networks, 255 nodes

CIDR = Classless Inter Domain Routing

- Eliminates class restrictions giving finer control to netmask.
- Uses 192.168.1.99/24 nomenclature (24 = # of ones from left)

IPv4 network classes

Your IP address identifies the “neighborhood” your node is in.

“Private IP addresses” are not assigned by the IANA (Internet Assigned Numbers Authority)

Addresses	CIDR Equivalent	Purpose	RFC	Class	Total # of addresses
0.0.0.0 - 0.255.255.255	0.0.0.0/8	Zero Addresses	RFC 1700	A	16,777,216
10.0.0.0 - 10.255.255.255	10.0.0.0/8	Private IP addresses	RFC 1918	A	16,777,216
127.0.0.0 - 127.255.255.255	127.0.0.0/8	Localhost Loopback Address	RFC 1700	A	16,777,216
169.254.0.0 - 169.254.255.255	169.254.0.0/16	Zeroconf	RFC 3330	B	65,536
172.16.0.0 - 172.31.255.255	172.16.0.0/12	Private IP addresses	RFC 1918	B	1,048,576
192.0.2.0 - 192.0.2.255	192.0.2.0/24	Documentation and Examples	RFC 3330	C	256
192.88.99.0 - 192.88.99.255	192.88.99.0/24	IPv6 to IPv4 relay Anycast	RFC 3068	C	256
192.168.0.0 - 192.168.255.255	192.168.0.0/16	Private IP addresses	RFC 1918	C	65,536
198.18.0.0 - 198.19.255.255	198.18.0.0/15	Network Device Benchmark	RFC 2544	C	131,072
224.0.0.0 - 239.255.255.255	224.0.0.0/4	Multicast	RFC 3171	D	268,435,456
240.0.0.0 - 255.255.255.255	240.0.0.0/4	Reserved	RFC 1700	E	268,435,456

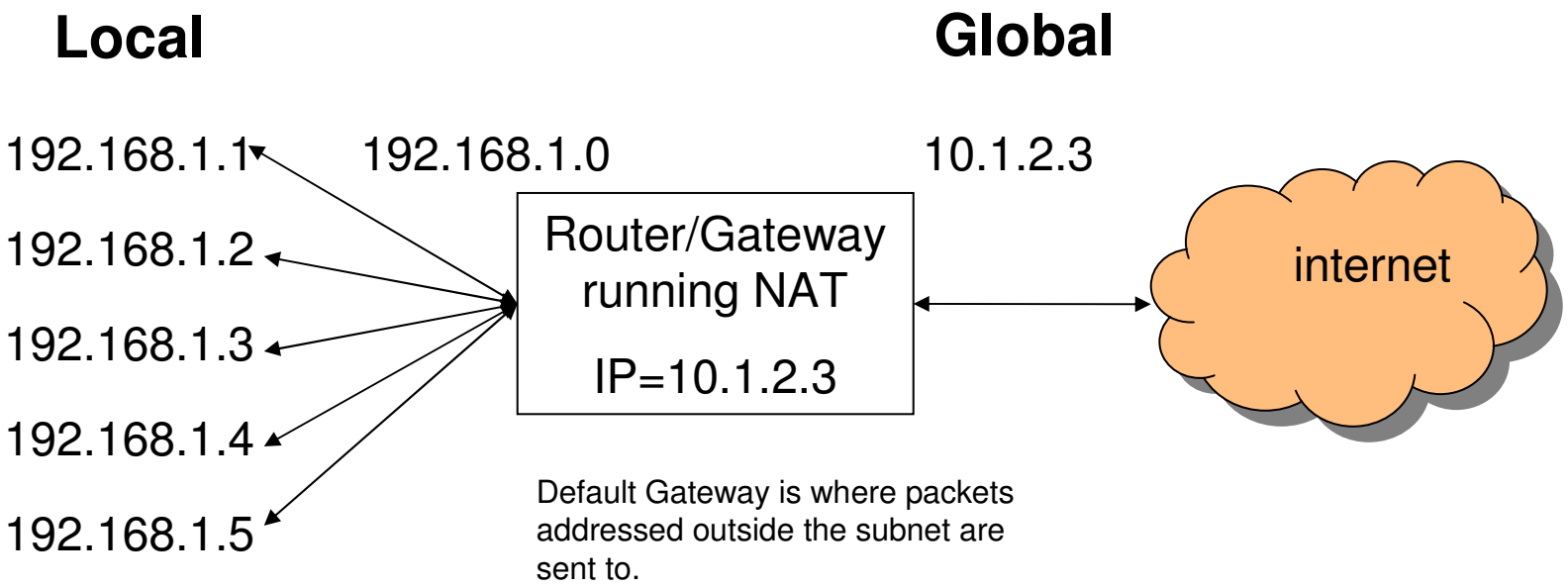
freescale semiconductor

NAT

Network Address Translation (**NAT**, also known as **network masquerading** or **IP-masquerading**) involves re-writing the source and/or destination addresses of IP packets as they pass through a router or firewall.

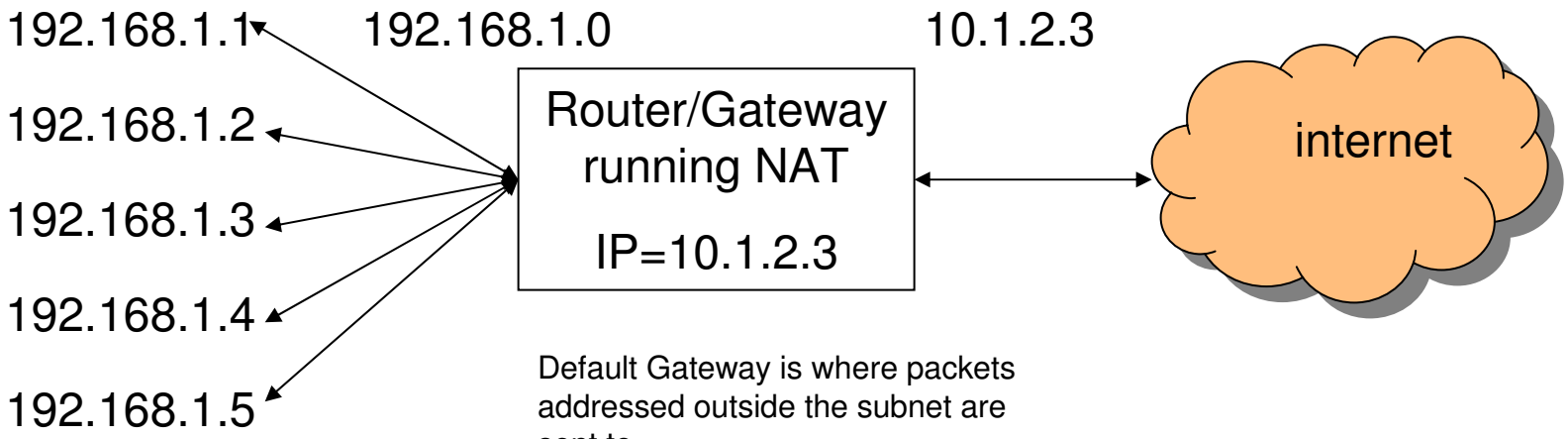
Most systems using NAT do so in order to enable multiple hosts on a private network to access the Internet using a single public IP address.

NAT is a non-standard protocol



freescale semiconductor

Default Gateway



Node 192.168.1.5 needs to send a packet to 207.68.172.246 (msn.com)

Node 192.168.1.5 identifies that 207.68.172.246 is outside the subnet (255.255.255.0)

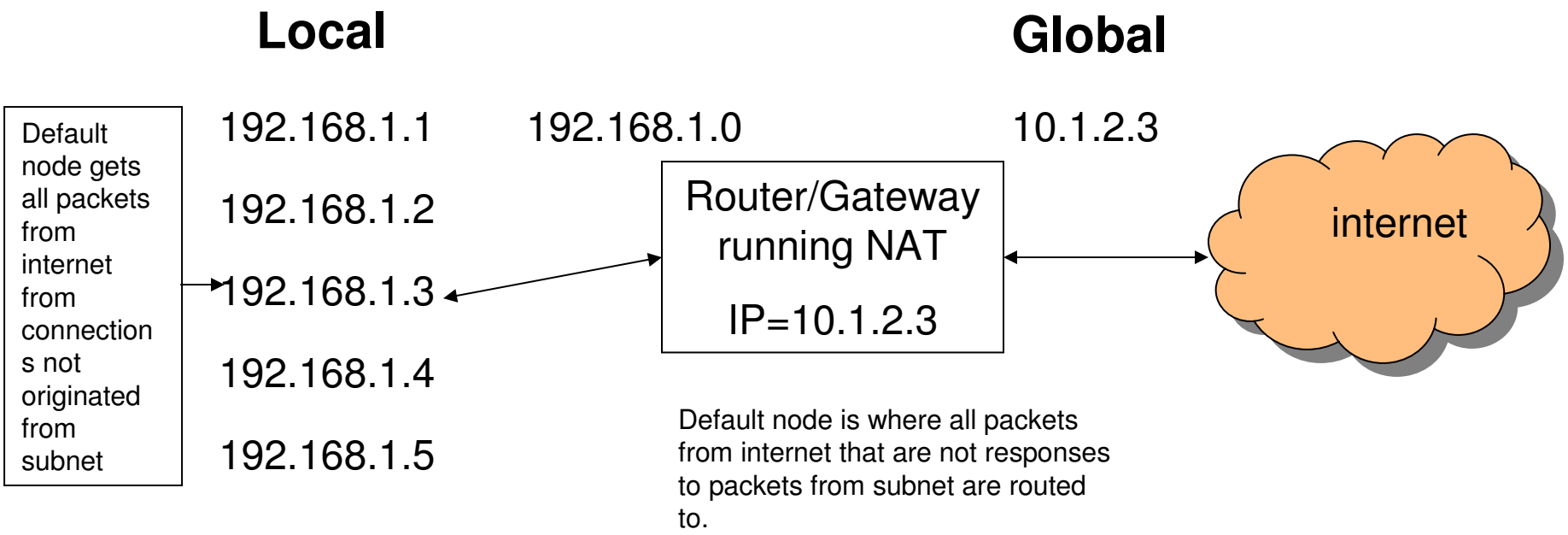
The packet is sent to 192.168.1.0, the default gateway

NAT translates the source field to 10.1.2.3

The internet sees a packet from 10.1.2.3 to 207.68.172.246

freescale semiconductor

Getting packets into a NAT network



If the connection originates from the internet (like connecting to a web server on an embedded device) NAT has a default node.

The default node is defined in the router/gateway.

Some routers/gateways always default to x.x.x.1

freescale semiconductor

TCP is One of the Protocols in the Internet Protocol Suite

TCP - Transport Control Protocol

TCP provides a “virtual” connection from one point to another.

The protocol **guarantees reliable and in-order delivery** of sender to receiver data. TCP also distinguishes data for multiple, concurrent applications (e.g. web server and email server) running on the same host.

TCP supports many of the Internet's most popular application protocols and resulting applications, including the world wide web, and email.

Other Protocols in the Internet Protocol Suite

HTTP - HyperText Transport Protocol - **Supported by Freescale Web Server**
- Used to transport HTML (web pages)

POP3 - Post Office Protocol
- Used to “pull” email from a server

TFTP - Trivial File Transfer protocol - **Supported by ColdFire Lite stack**
- Used to transfer blocks of data

UDP - User Datagram Protocol - **Supported via Interniche**
- Used to transfer data without a connection
- Provides 65535 multiplexed ports per IP address

PPP - Point to Point Protocol - **Supported via Interniche**
- used to establish a direct connection between two nodes

DHCP - Dynamic Host Configuration Protocol - **Supported by ColdFire Lite stack**
- Used to dynamically configure a device on a network

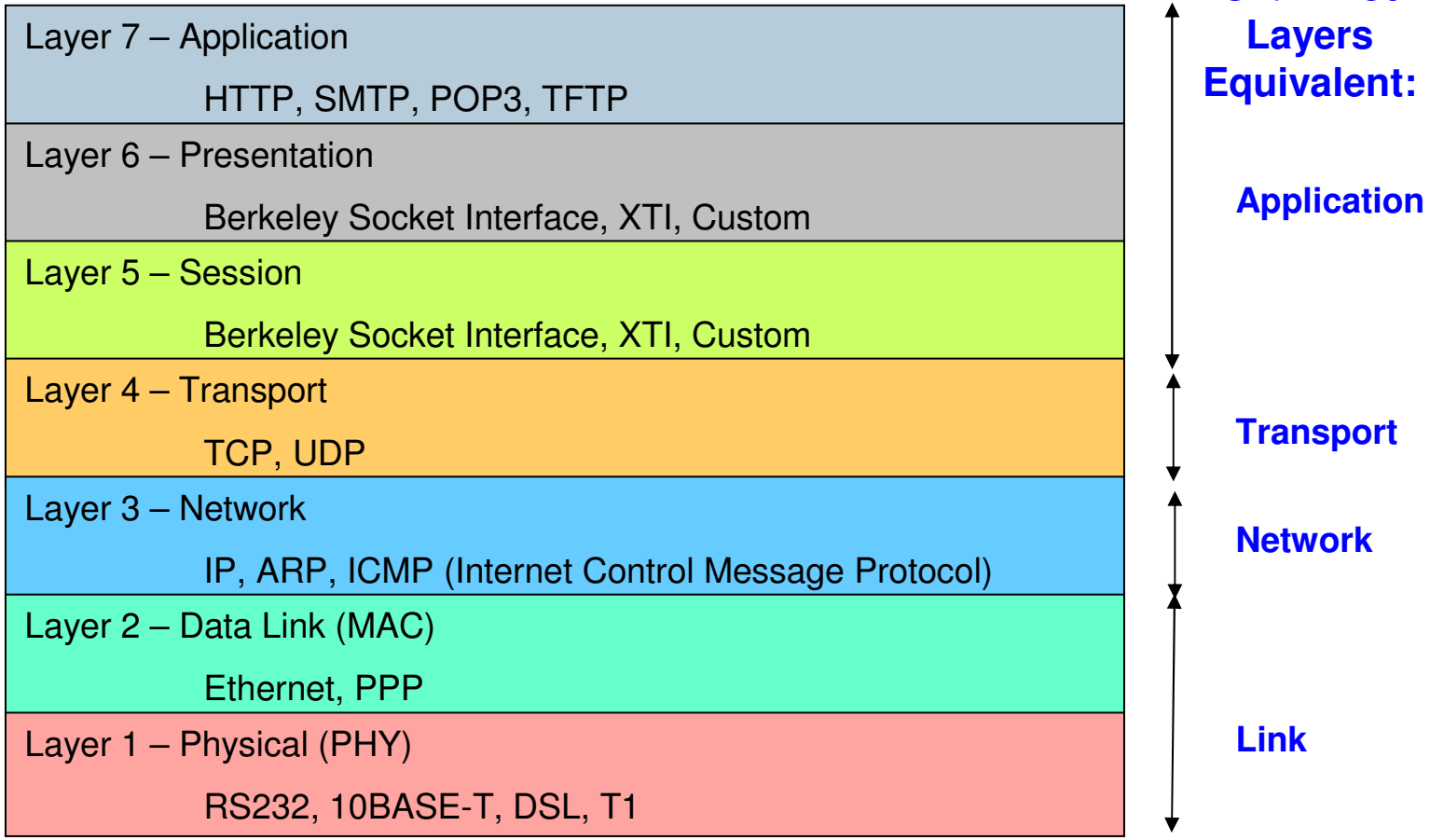
BOOTP – Bootstrap Protocol - **Supported by ColdFire Lite stack**
- Another much simpler method of dynamically configuring a device on a network

DNS – Dynamic Name System - **Supported via Interniche**
- A client/server based system to translate host/domain names or URL’s to IP addresses.

ARP – Address Resolution Protocol - **Supported by ColdFire Lite stack**
- A lower level protocol required to match IP addresses and Ethernet MAC addresses

The OSI 7 Layer Model

The **Open Systems Interconnection Reference Model** (**OSI Model** or **OSI Reference Model** for short) is a layered abstract description for communications and computer network protocol design, developed as part of the Open Systems Interconnect initiative.



All People Seems To Need Data Processing".

freescale semiconductor

Some Interesting RFC's

0008 Functional specifications for the ARPA Network. G. Deloche. May-05-1969. (Not online) (Status: UNKNOWN)

0009 Host software. G. Deloche. May-01-1969. (Not online) (Status: UNKNOWN)

0011 Implementation of the Host-Host software procedures in GORDO. G. Deloche. Aug-01-1969. (Not online) (Obsoleted by RFC0033) (Status: UNKNOWN)

0015 Network subsystem for time sharing hosts. C.S. Carr. Sep-25-1969. (Format: TXT=10695 bytes) (Status: UNKNOWN)

0016 M.I.T. S. Crocker. Aug-27-1969. (Format: TXT=682 bytes) (Obsoletes RFC0010) (Obsoleted by RFC0024) (Updated by RFC0024, RFC0027, RFC0030) (Status: UNKNOWN)

0017 Some questions re: Host-IMP Protocol. J.E. Kreznar. Aug-27-1969. (Format: TXT=6065 bytes) (Status: UNKNOWN)

0018 IMP-IMP and HOST-HOST Control Links. V. Cerf. Sep-01-1969. (Format: TXT=634 bytes) (Status: UNKNOWN)

0019 Two protocol suggestions to reduce congestion at swap bound nodes. J.E. Kreznar. Oct-07-1969. (Format: TXT=3392 bytes) (Status: UNKNOWN)

0020 ASCII format for network interchange. V.G. Cerf. Oct-16-1969. (Format: TXT=18504 bytes) (Status: UNKNOWN)

0021 Network meeting. V.G. Cerf. Oct-17-1969. (Format: TXT=2143 bytes) (Status: UNKNOWN)

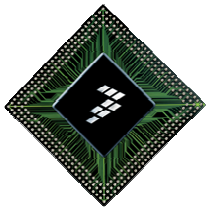
0022 Host-host control message formats. V.G. Cerf. Oct-17-1969. (Format: TXT=4606 bytes) (Status: UNKNOWN)

0023 Transmission of Multiple Control Messages. G. Gregg. Oct-16-1969. (Format: TXT=690 bytes) (Status: UNKNOWN)

0031 Binary Message Forms in Computer. D. Bobrow, W.R. Sutherland. Feb-01-1968. (Format: TXT=11191 bytes) (Status: UNKNOWN)

0033 New Host-Host Protocol. S.D. Crocker. Feb-12-1970. (Format: TXT=44167 bytes) (Obsoletes RFC0011) (Updated by RFC0036, RFC0047) (Status: UNKNOWN)

Freescale solution for Ethernet



freescale™
semiconductor

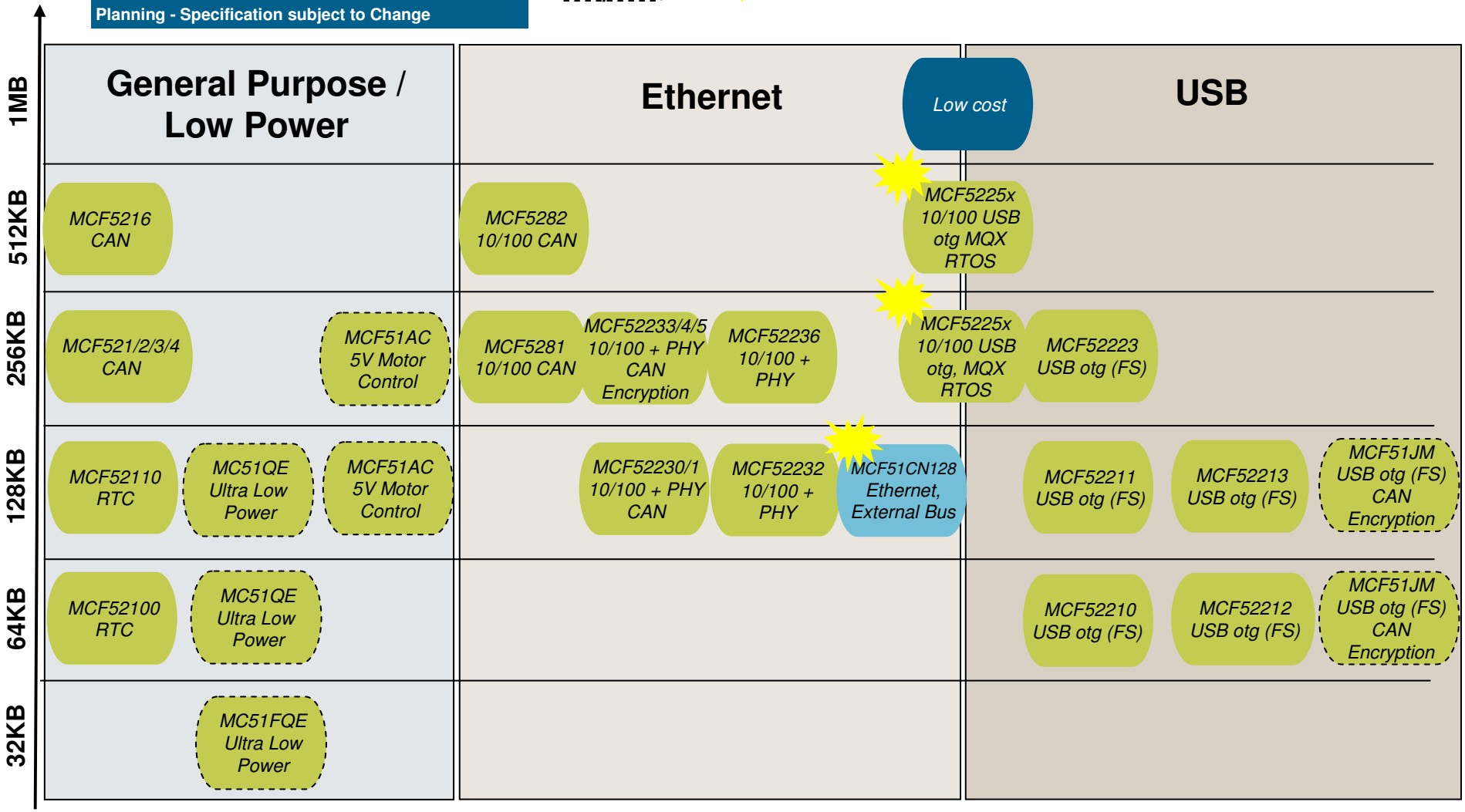
ColdFire MCU Roadmap

Flash

Production - Available NOW
Execution - Specification frozen, in design
Planning - Specification subject to Change

Flexis 8-bit Compatible

2009

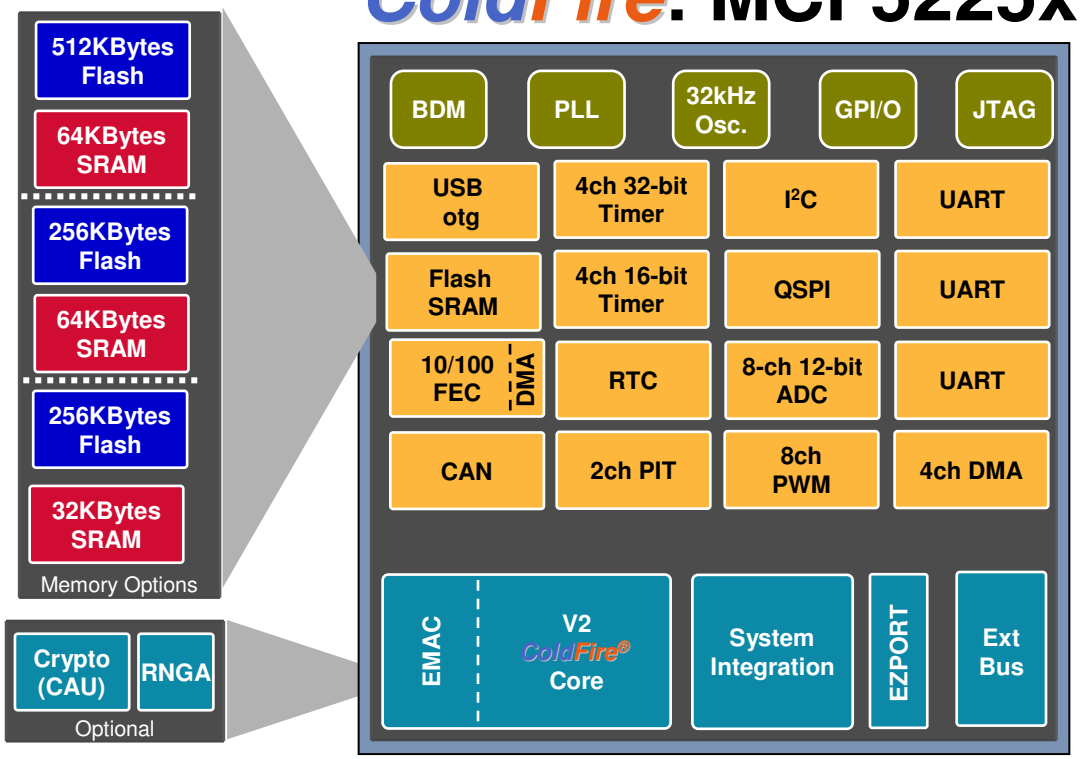


freescale semiconductor

ColdFire: MCF5225x

ColdFire V2 Core

- Up to 76 Dhrystone 2.1 MIPS @ 80 MHz
- MAC Module and HW Divide
- **Encryption – CAU**
- **External Bus**
- Up to 64K bytes SRAM
- **Up to 512K bytes Flash**
- **USB 2.0 full-speed Host/Device/On-the-go Controller**
- **CAN – (FlexCAN)**
- **FEC (10/100 Ethernet)**
- 3 UARTs
- Serial Peripheral Interface (Queued SPI)
- I^2C bus interface modules
- 4 ch. 32-bit timers with DMA support
- 4 ch. 16-Bit Capture/Compare/PWM timers
- 2 ch. Periodic Interrupt Timer
- 8 ch. PWM timer with enhanced DAC capabilities
- 2nd Watchdog timer with independent clock
- Real Time Clock with 32kHz crystal oscillator
- 8 ch. 12-bit A-to-D converter with simultaneous sampling
- Up to 56 5V Tolerant General-Purpose I/O
- System Integration (PLL, SW Watchdog)



- Single 3.3V supply
- Temperature Range: -40°C to +85°C
- Available Speeds: 66 and 80MHz
- Available packages: 100 LQFP, 144 LQFP, 144 BGA

freescale semiconductor

MCF52259 –FEC features

The FEC incorporates the following features:

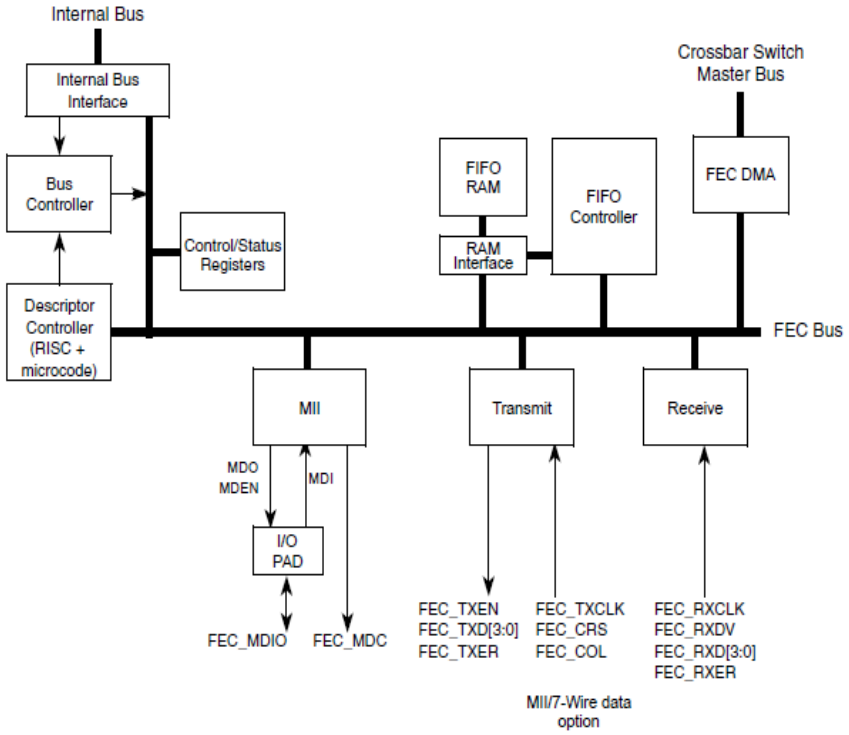
- Support for three different Ethernet physical interfaces:
  - 100-Mbps IEEE 802.3 MII
  - 10-Mbps IEEE 802.3 MII
  - 10-Mbps 7-wire interface (industry standard)
- IEEE 802.3 full duplex flow control
- Programmable max frame length supports IEEE 802.1 VLAN tags and priority
- Support for full-duplex operation (200 Mbps throughput) with a minimum internal bus clock rate of 50 MHz
- Support for half-duplex operation (100 Mbps throughput) with a minimum internal bus clock rate of 50 MHz
- Retransmission from transmit FIFO following a collision (no processor bus utilization)
- Automatic internal flushing of the receive FIFO for runts (collision fragments) and address recognition rejects (no processor bus utilization)
- Address recognition
  - Frames with broadcast address may be always accepted or always rejected
  - Exact match for single 48-bit individual (unicast) address

MCF52259-FEC Fetures

- Hash (64-bit hash) check of individual (unicast) addresses
- Hash (64-bit hash) check of group (multicast) addresses
- Promiscuous mode

MCF522xx – Ethernet Media Access Controller (MAC)

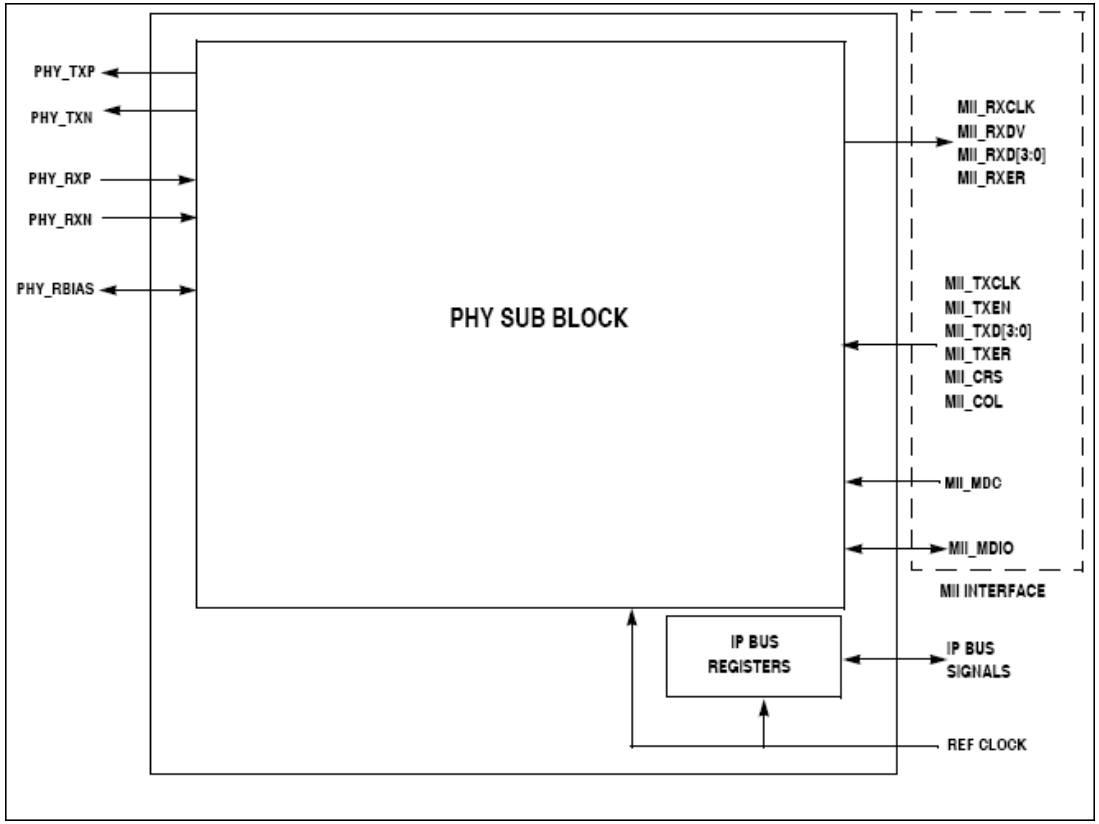
- The Ethernet MAC supports 10/100 Mbps Ethernet/IEEE 802.3 networks
- IEEE 802.3 full duplex flow control
- Support for full-duplex operation (40Mbps throughput) with a minimum system clock rate of 50MHz
- Support for half-duplex operation (20Mbps throughput) with a minimum system clock rate of 25MHz



freescale semiconductor

MCF5223x - ePHY

- The ePHY (embedded PHYsical layer interface) is IEEE 802.3 compliant
- Supports both the media-independent interface (MII) and the MII management interface
- Full-/half-duplex support in all modes
- Requires a 25-MHz crystal for its basic operation
- Supports Loopback modes



freescale semiconductor

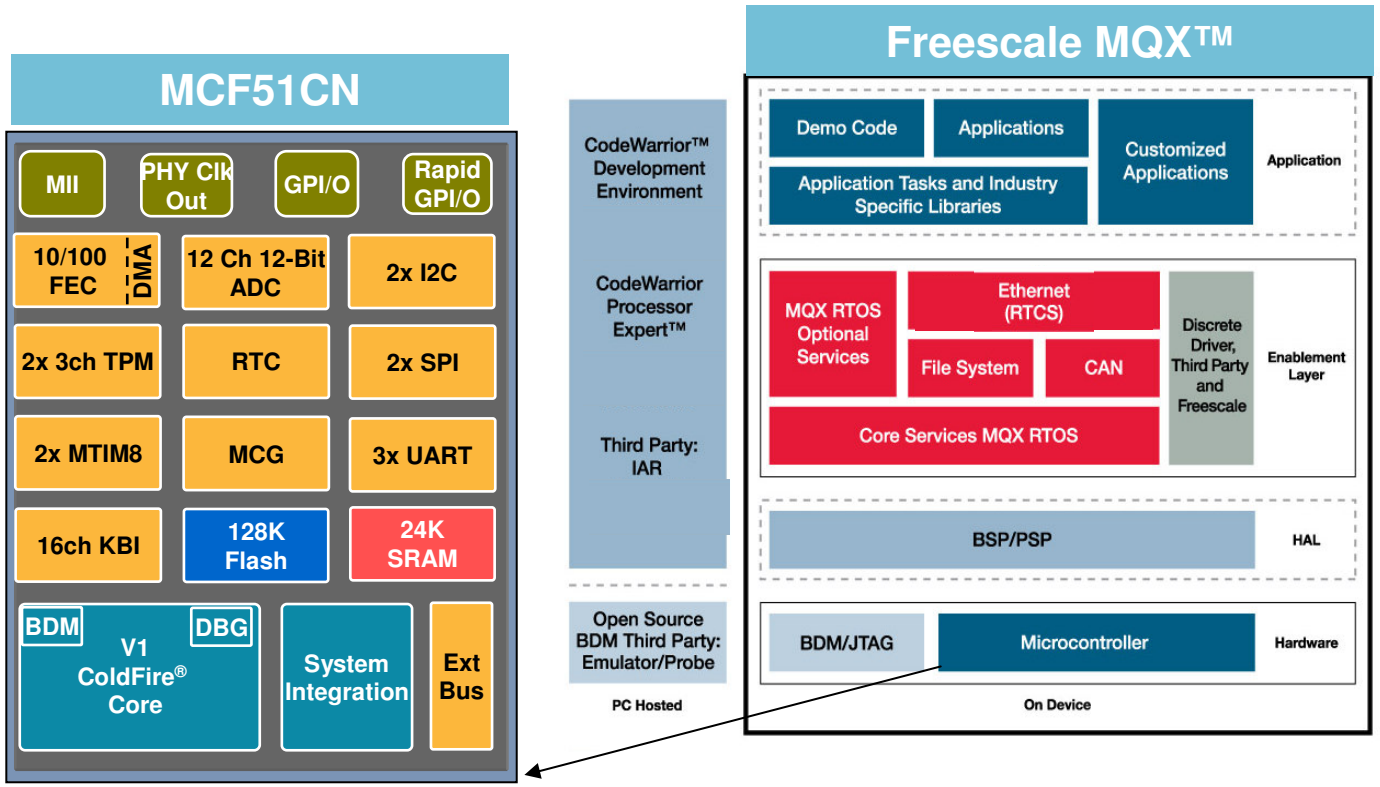
Complete Ethernet Solution

MCF51CN

- Up to 46 Dhrystone 2.1 MIPS @ 50 MHz
- Mini-FlexBus support up to 2MB external memory
- Ethernet:
  - 10/100 FEC – Fast Ethernet Controller with DMA
  - MII Interface with Output Clock for PHY
  - Support Half/Full Duplex

MQX Software

- Reuse of software
- Full production source code
- Developers keep their source modifications
- Small, configurable footprint
- Integrated communication suite (RTCS)
- Eliminates initial software investment hurdle
- $95K worth of software from day one



www.freescale.com/mqx

freescale™ semiconductor

MCF51CN128

68K/ColdFire® V1 Core

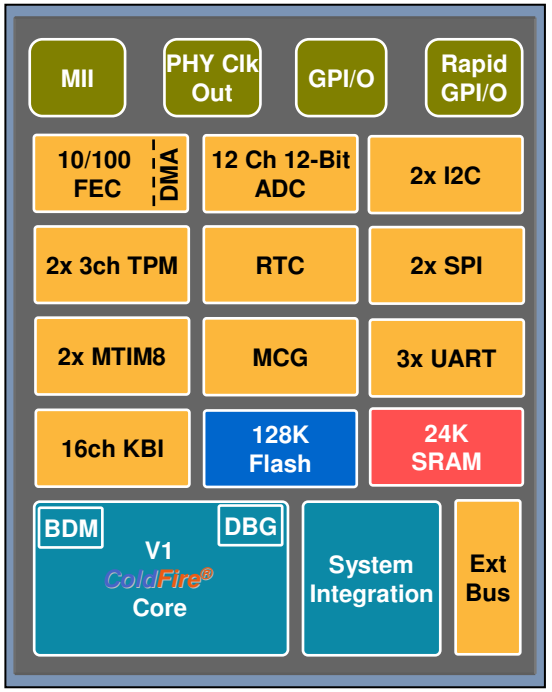
- Up to 46 Dhrystone 2.1 MIPS @ 50 MHz
- Mini Flexbus support up to 1MB external memory (80LQFP) support 2 Devices

Memory

- 128K bytes flash
- 24K bytes SRAM

Features

- Ethernet:
  - 10/100 FEC – Fast Ethernet Controller with DMA
  - MII Interface with Output Clock for PHY
  - Support Half/Full Duplex
- Low power mode – Ethernet operation supported at 3V and above
- Ultra-small (7x7mm) 48-pin package
- 12-Ch, 12-Bit ADC
- 3x UARTs (2 on 48 pin, 3 on 64/80 pin)
- 2x SPI
- 2x I^2C bus interface
- Real Time Counter
- Up to 70 General-Purpose I/O
- System Integration (PLL, SW Watchdog)
- Single Voltage Supply 1.8-3.6V



Part #	Package	ADC Ch	KBI	Port I/O	Rapid GPIO	SCI (UART)	TPM Ch	Ext Bus lines addr/data/chip select	10K# SRP
MCF51CN128CLK	80LQFP	12	16	70	16	3	6	20 / 8 / 2	$3.31
MCF51CN128CGT	64LQFP	12	12	54	16	3	6	-	$3.21
MCF51CN128CLH	48QFN	12	6	38	8	3	6	-	$2.99

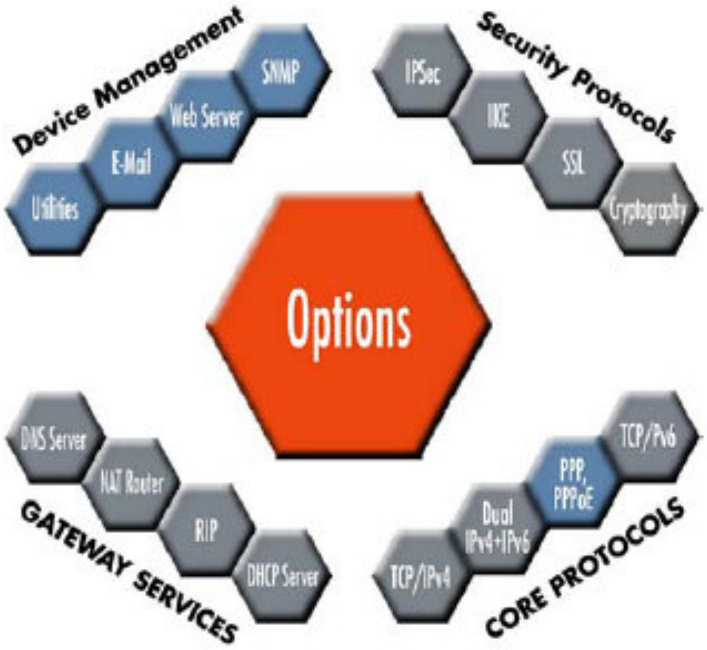
freescale semiconductor

Freescale Complimentary Software Solution

InterNiche and Freescale have collaborated to provide and OEM version of InterNiche's NicheLite optimized for the ColdFire architecture.

Key Features:

Address Resolution Protocol (ARP)
Internet Protocol (IP)
Internet Control Messaging Protocol (ICMP)
User Datagram Protocol (UDP)
Transmission Control Protocol (TCP)
Dynamic Host Configuration Protocol (DHCP) Client
Bootstrap Protocol (BOOTP)
Trivial File Transfer Protocol (TFTP)



Options

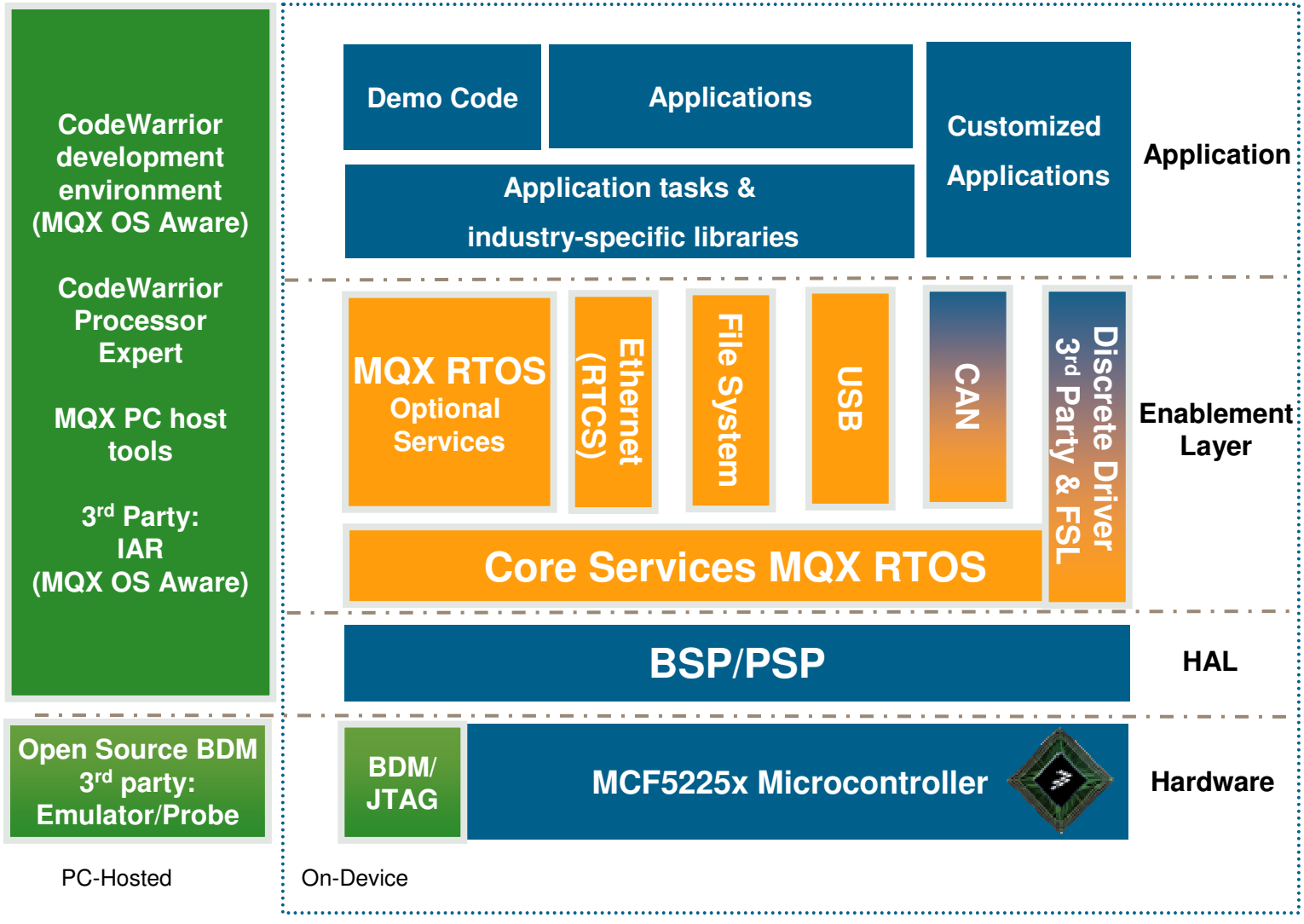
Email
FTP
HTTP Server
PPP
PPPoE
SNMP
Telnet

Upgrades

NicheStack v4, v6, v4/v6
Security modules
Cryptography
Routing/gateway
DHCP Server
DNS Server

freescale™ semiconductor

Freescale Complete Solution



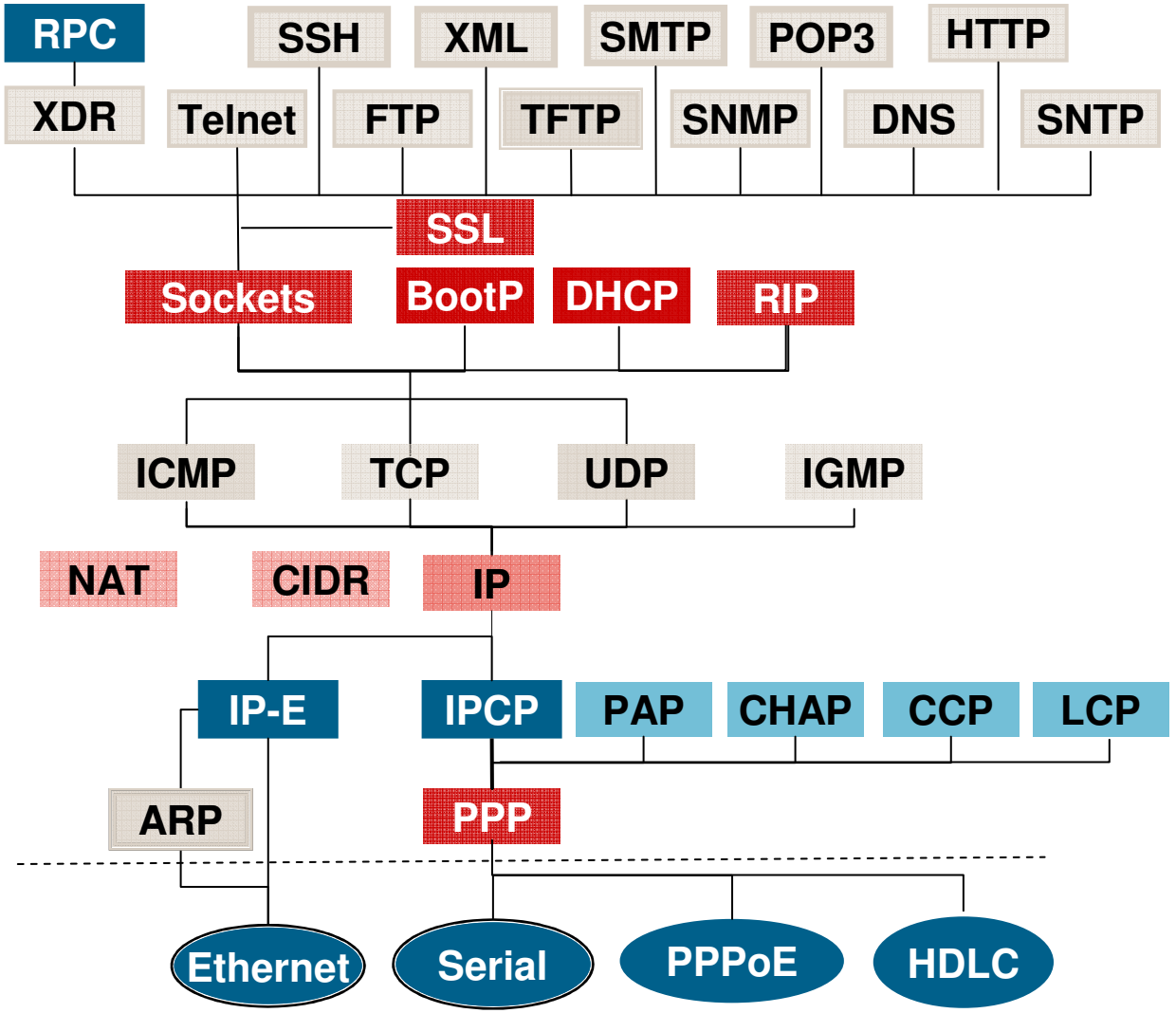
- MCF5225x is a one-stop-shop connectivity MCU with on-chip USB, Ethernet, CAN and encryption.
- MQX offers RTOS, USB Stack and drivers, RTCS of Ethernet stack, file systems and more functions
- A single CodeWarrior development studio supports the entire ColdFire Family.
- Processor Expert allows GUI-enabled interface.
- Cost effective and full function development tools with built in BDM module for in-circuit debug

Freescale Enablement (MQX)

freescale semiconductor

RTCS – Real-time TCP/IP Communications Suite

Protocols



freescale semiconductor

Freescale MQX Software Solutions

Freescale owns

- Source code, rights to distribute and modify across the Freescale Portfolio

Benefits

- Full production source code* with silicon
- Commercial-friendly licensing model that lets developers keep their source modifications
- Small, configurable footprint
- Integrated stacks (TCP/IP, USB, etc.)

Value

- Eliminates initial software investment hurdle
- $95K worth of software from day one

Proven

- Market-proven on Freescale processors for over 15 years and used in millions end use products, now a part of the offering.

One Collaborative Source

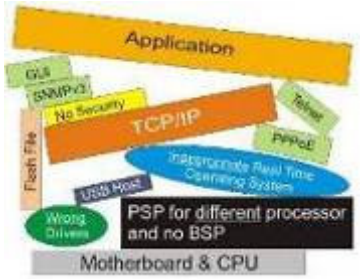
- Hardware: MCF5225x (ColdFire)
- Tool: CodeWarrior ™(CW7.1 plus v7.1.1) & IAR System (RTOS Task aware debugging)
- Run-time software: Freescale MQX
- Strong Third Party Support Network

Full- Featured and Powerful

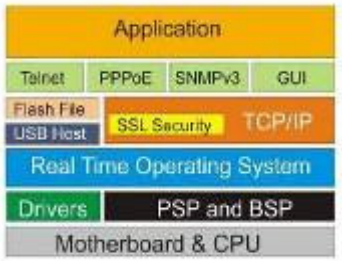
What is Freescale MQX?

- **RTOS (Full priority-based, pre-emptive scheduler)**
- **Real-time TCP/IP Communication Suite (RTCS)**
  - TCP/IP, FTP, Telnet, DHCP, SNMP etc..
- **USB Host -** HID, MASS, HUB
- **USB Device** - HID, MASS, CDC
- **MS-DOS File System (MFS)**
- **BSP I/O Driver: CAN, UART etc…**
- **HTTP Web server**

Past Customer Problem



The Solution



* Complimentary with MCF5225x. Subject to License Agreement

freescale ™ semiconductor

Coldfire TCP/IP Stack Features

- HTTP server
- HTTP client
- RSS/XML client
- TCP/UDP client and server
- Serial to Ethernet client and server
- TFTP
- DHCP or Manual IP configuration
- Domain name server client (DNS)

- Transmission control protocol (TCP)
- User Datagram protocol (UDP)
- Internet controlling message protocol (ICMP)
- BOOT strap protocol
- Address resolution protocol (ARP)
- Internet protocol (IP)

TCP/IP stack overview

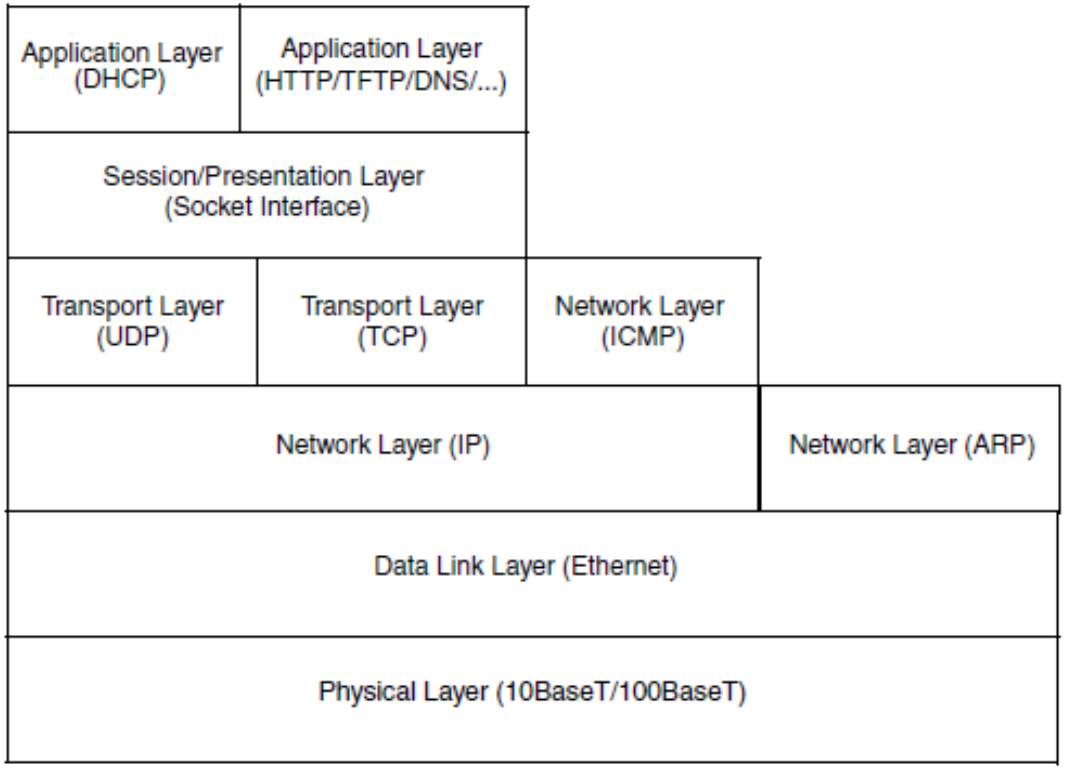


Figure 6. TCP/IP Stack Overview

freescale™ semiconductor

TCP/IP stack structure

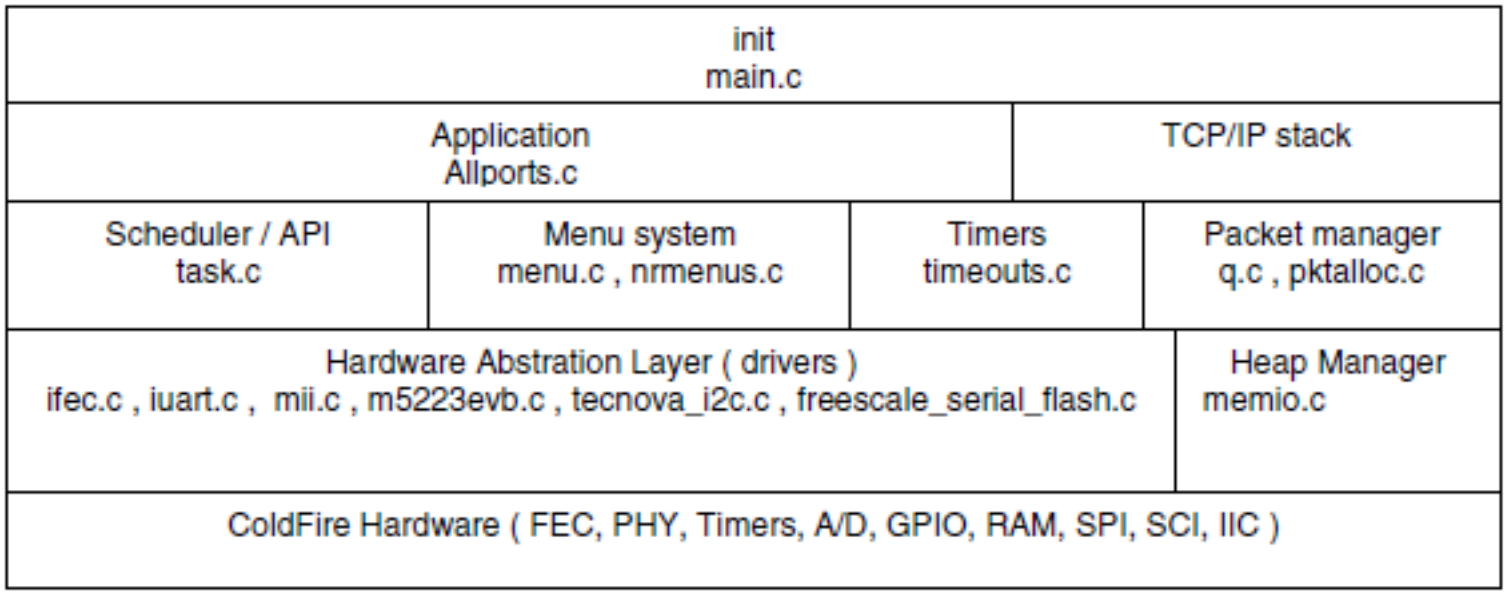


Figure 1. ColdFire TCP/IP Stack

freescale™ semiconductor

More details about the stack

The TCP/IP stack implements the protocols described in the following RFC's (refer to http://www.rfc-editor.org/rfcxx00.html for details):

- RFC791: Internet protocol (IP)
- RFC792: Internet Control Message protocol (ICMP)
- RFC768: User Datagram Protocol (UDP)
- RFC793: Transmission Control Protocol (TCP)
- RFC826: Ethernet Address Resolution Protocol (ARP)
- RFC1035: Domain Name Server (DNS)
- RFC2131: Dynamic Host Configuration Protocol
- RFC2132: DHCP options

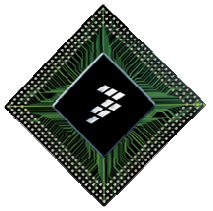
Coldfire TCP/IP flash and RAM requirement

Table 1. Flash and RAM Requirements for Various ColdFire TCPIP Builds

Target	Flash (bytes)	BSS+DATA (bytes)	Stack (bytes)	Heap (bytes)	Total RAM (bytes)
Stack Only	33744	2820	1024	7852	11696
UDP Client	34368	2856	1024	8870	12750
TCP Client	35344	2938	1024	8870	12832
UDP Server	34176	2856	1024	8870	12730
TCP Server	35520	3202	1024	8870	13096
TCP Serial Server	36176	3198	1024	8870	13092
TCP Serial Client	36256	3198	1024	8870	13092
Web Server	45264	4660	1024	9894	15578

freescale™ semiconductor

Ethernet LABs



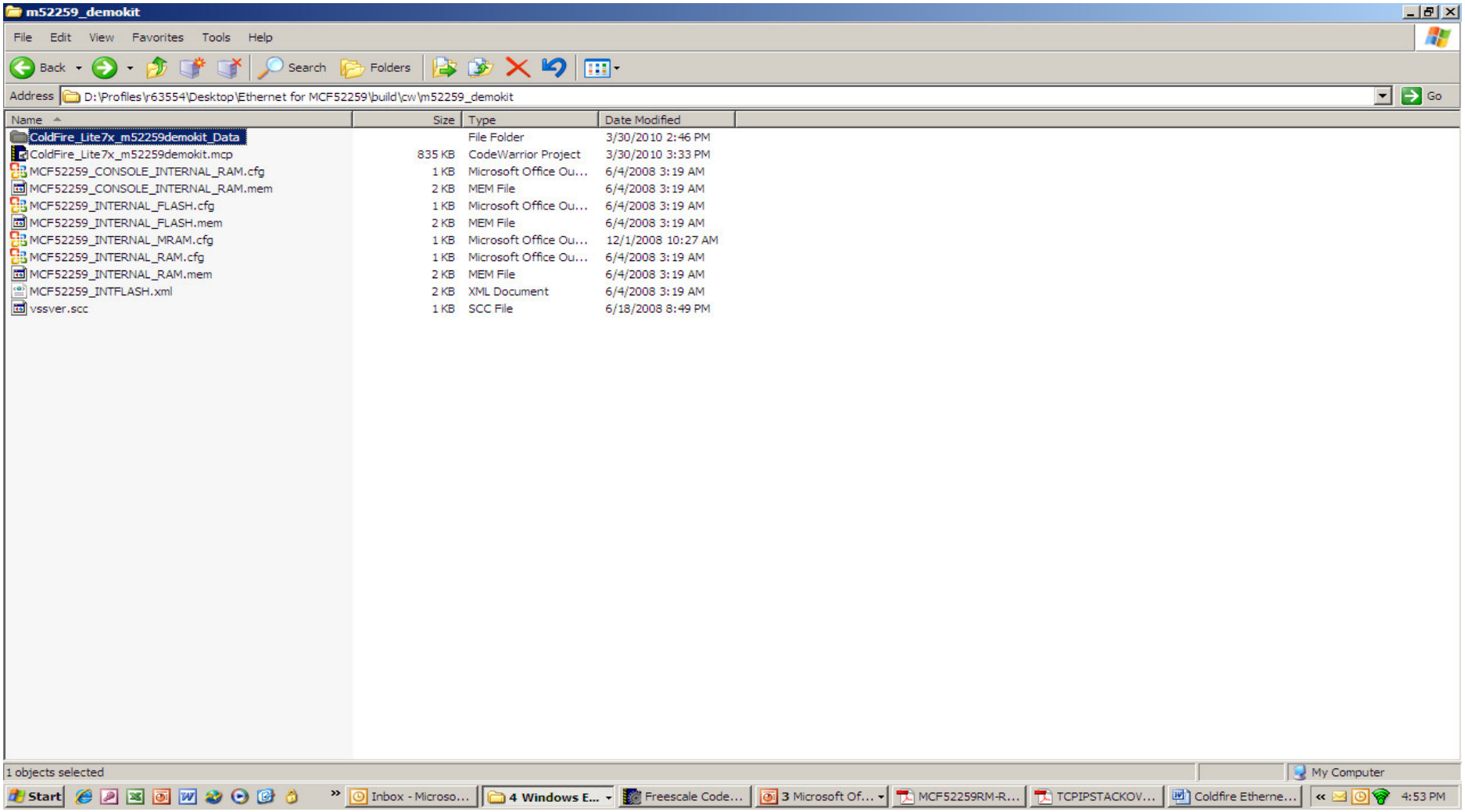
freescale™
semiconductor

LAB1

Install the Given exe file TCP_IPLite_MCF5225xI.exe

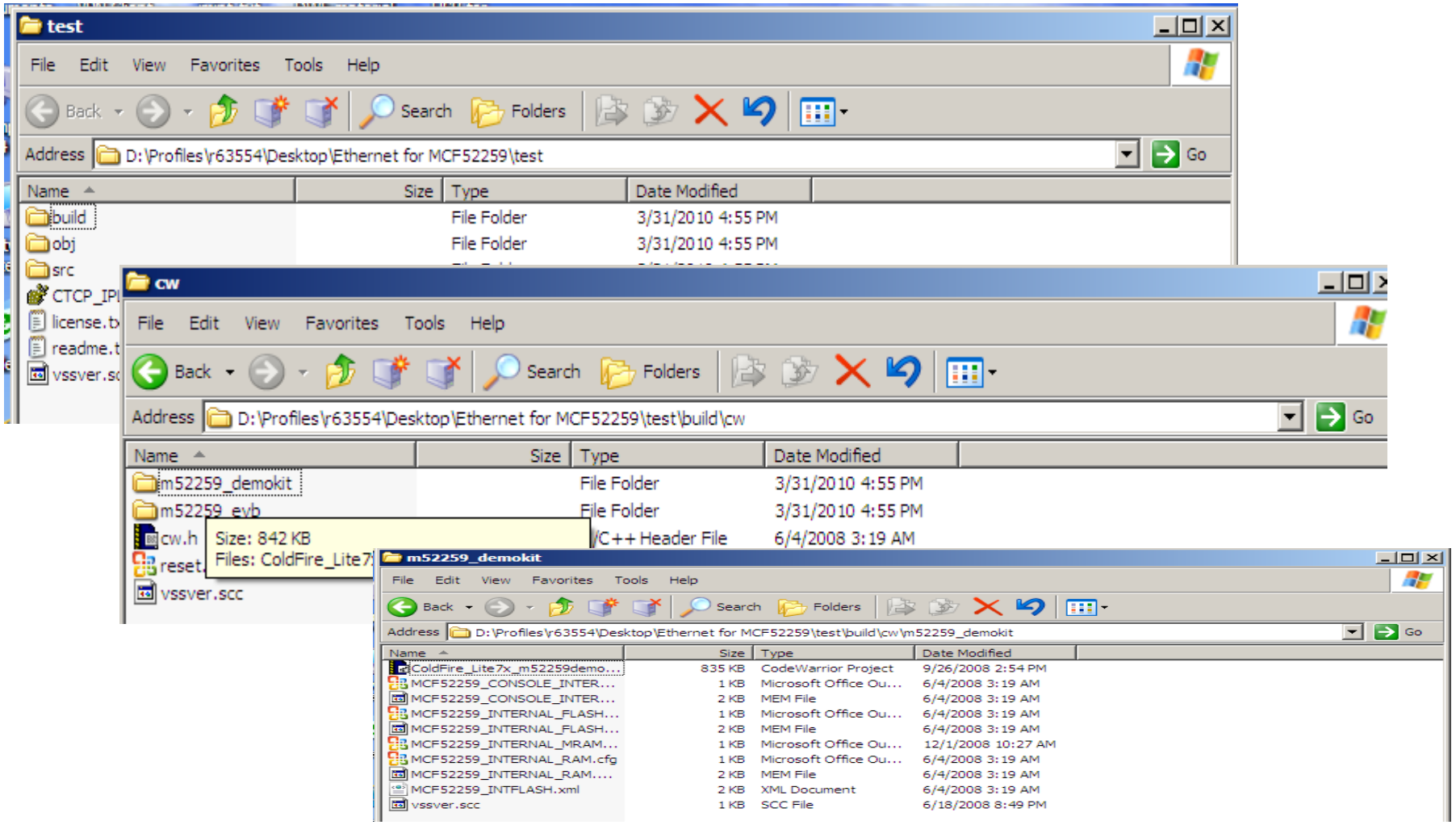
freescale ™
semiconductor

Directory Structure



freescale™ semiconductor

Directory Structure



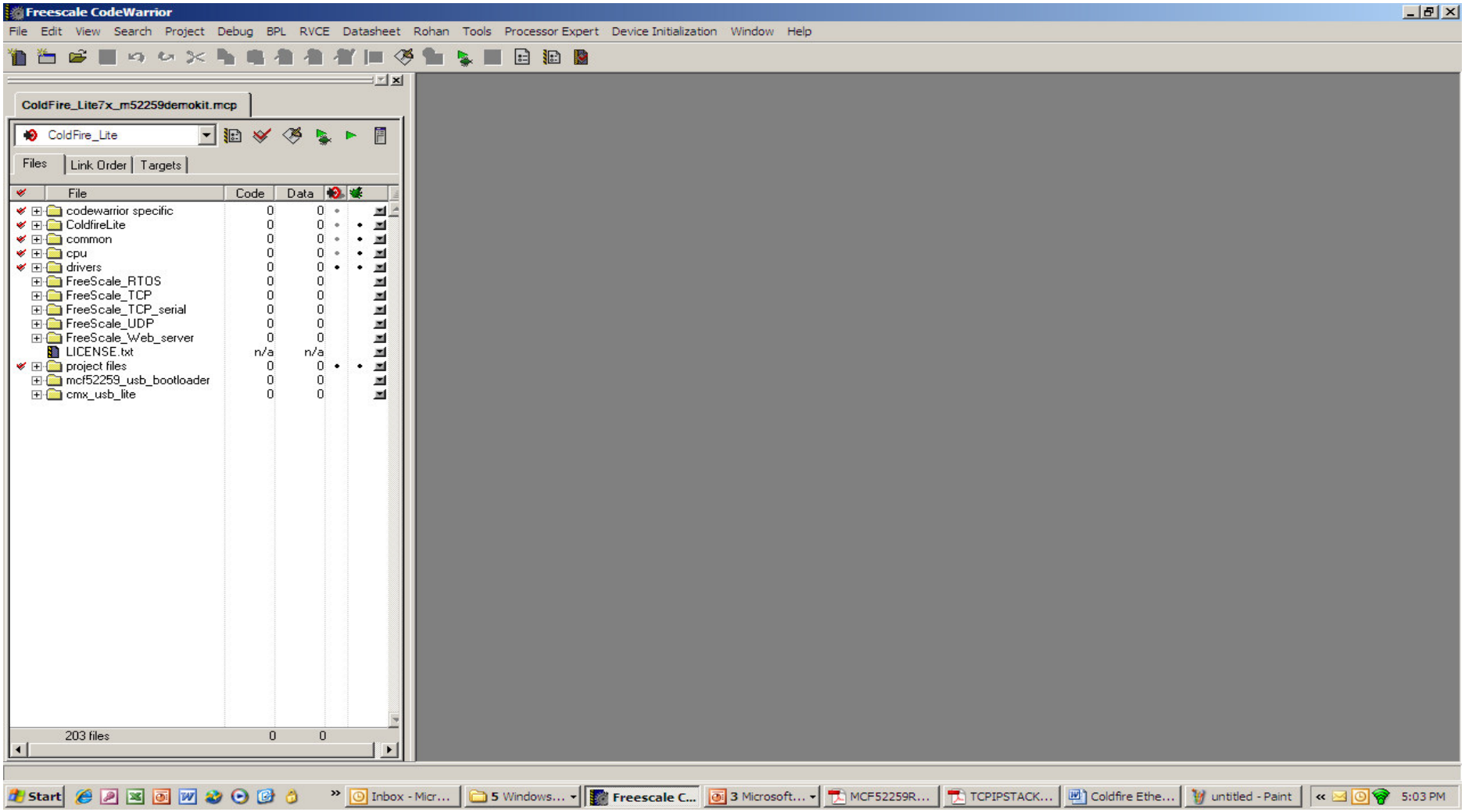
freescale semiconductor

Project file

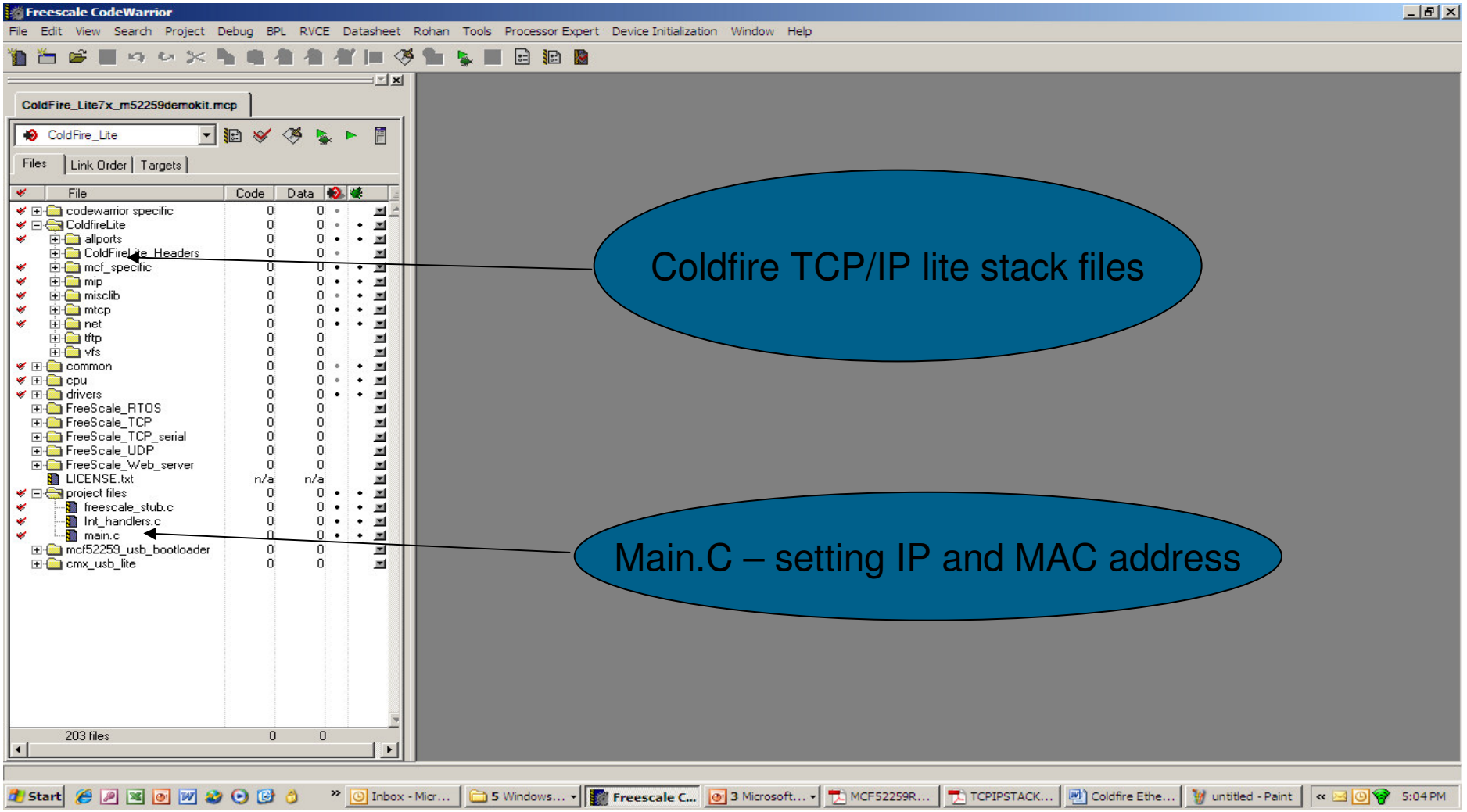
Open the codewarrior for coldfire

Open the Coldifre_Lite7x …*.mcp project file.

Project file details

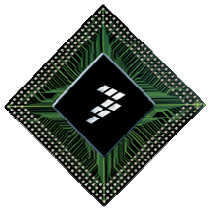


freescale™ semiconductor



freescale semiconductor

LAB1



freescale™
semiconductor

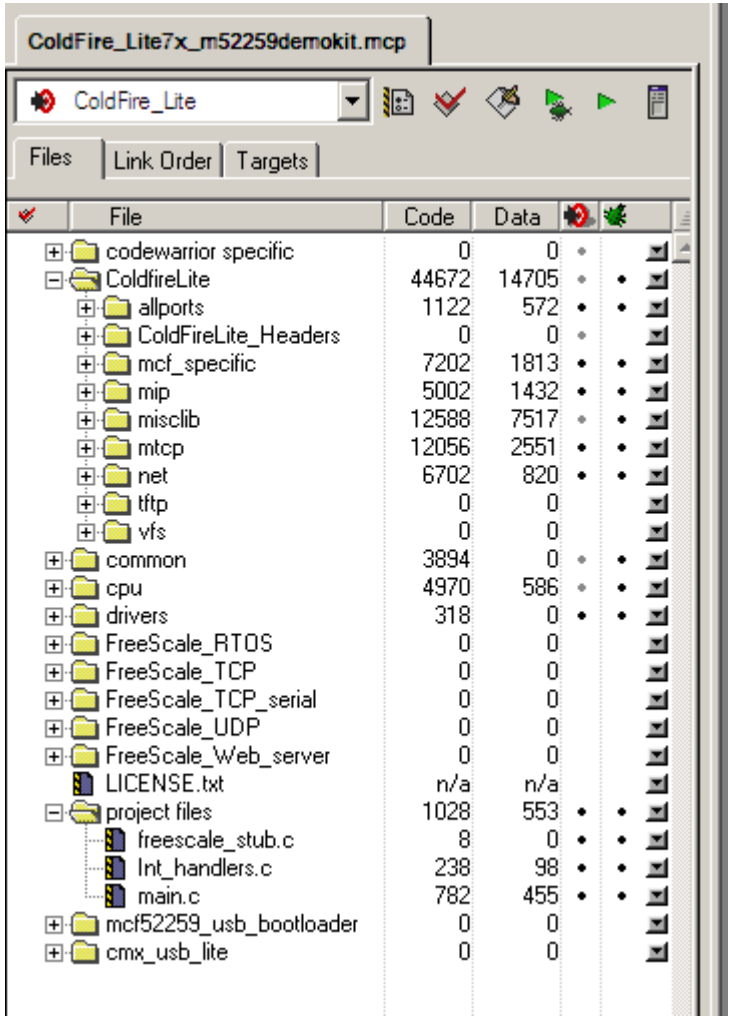
Flashing and booting the board

Connect the board via USB and serial to the board

Select the Coldfire_Lite project in code warrior.

Compile the code.

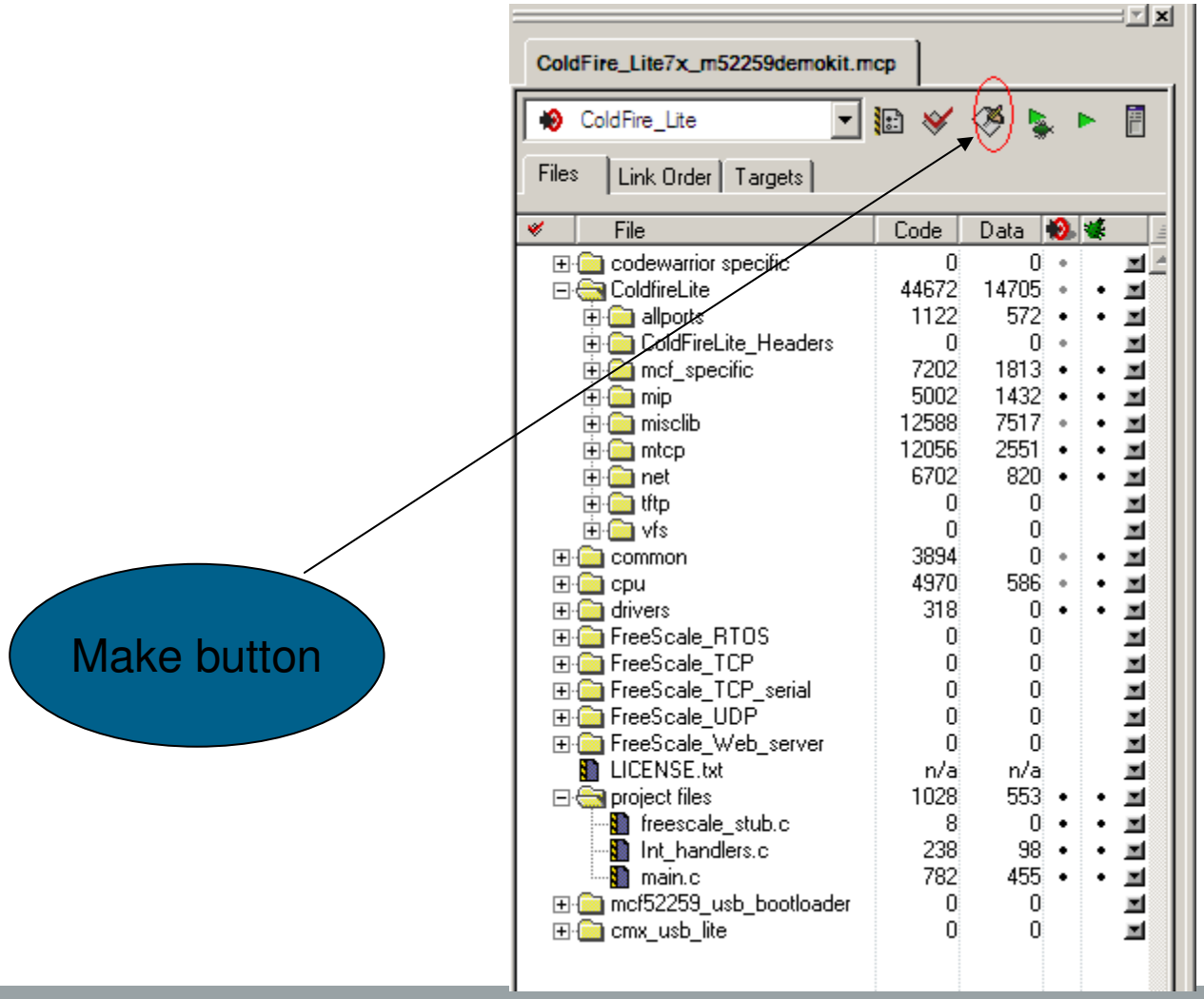
Flash the program into the controller using codewarrior



File	Code	Data
codewarrior specific	0	0
ColdfireLite	44672	14705
allports	1122	572
ColdFireLite_Headers	0	0
mcf_specific	7202	1813
mip	5002	1432
misclib	12588	7517
mtcp	12056	2551
net	6702	820
tftp	0	0
vfs	0	0
common	3894	0
cpu	4970	586
drivers	318	0
FreeScale_RTOS	0	0
FreeScale_TCP	0	0
FreeScale_TCP_serial	0	0
FreeScale_UDP	0	0
FreeScale_Web_server	0	0
LICENSE.txt	n/a	n/a
project files	1028	553
freescale_stub.c	8	0
Int_handlers.c	238	98
main.c	782	455
mcf52259_usb_bootloader	0	0
cmx_usb_lite	0	0

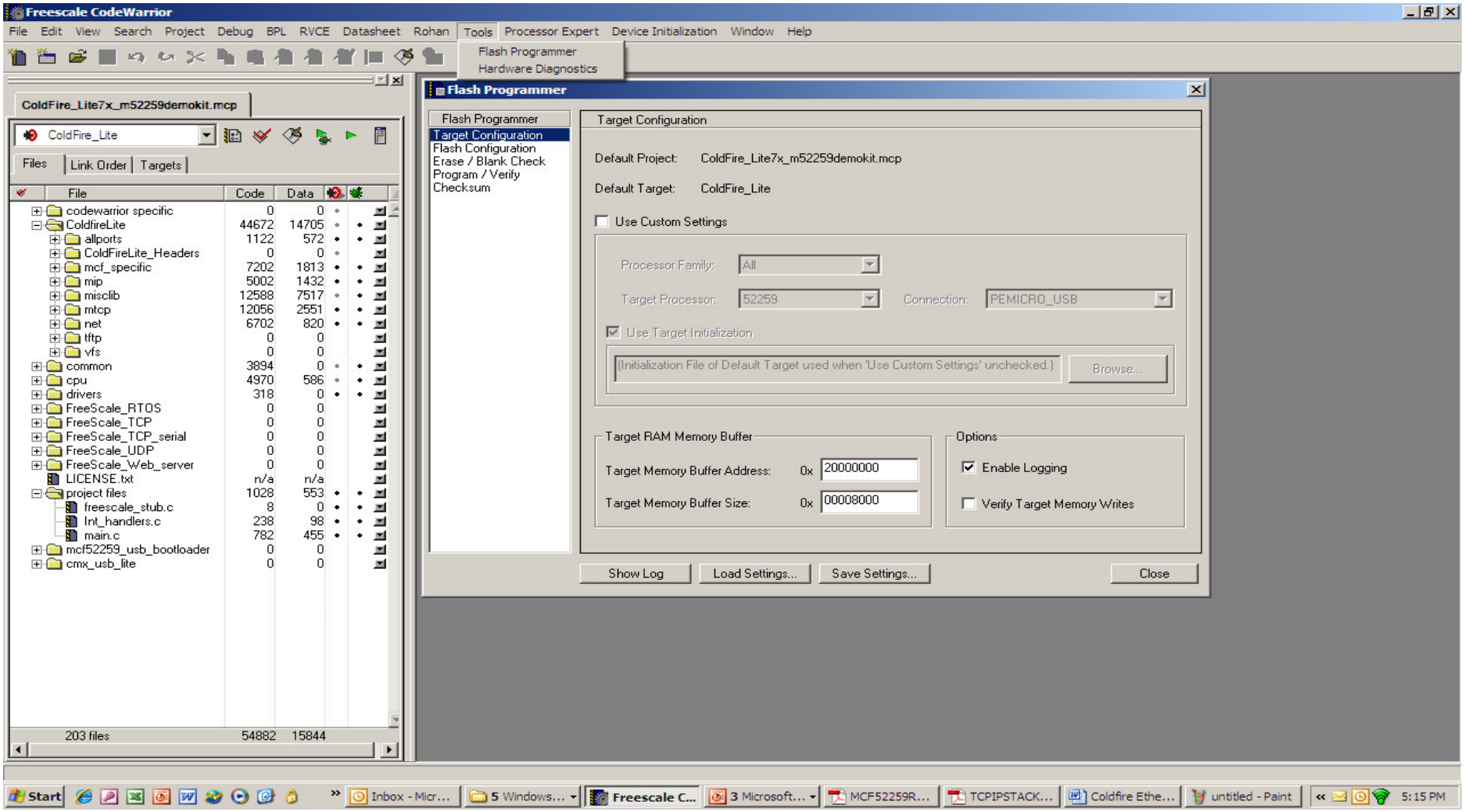
freescale™ semiconductor

Compiler the code



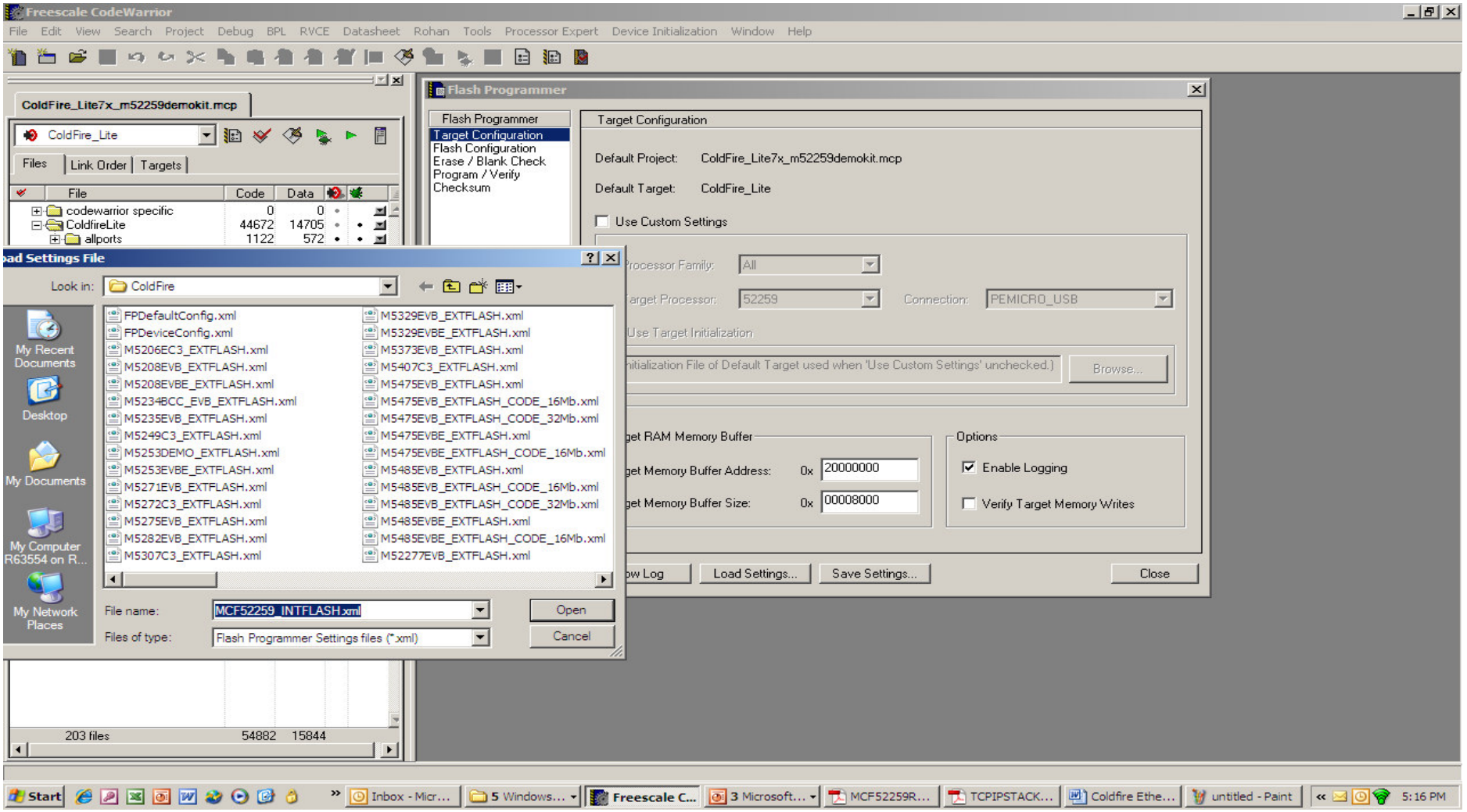
freescale semiconductor

Flash progamming



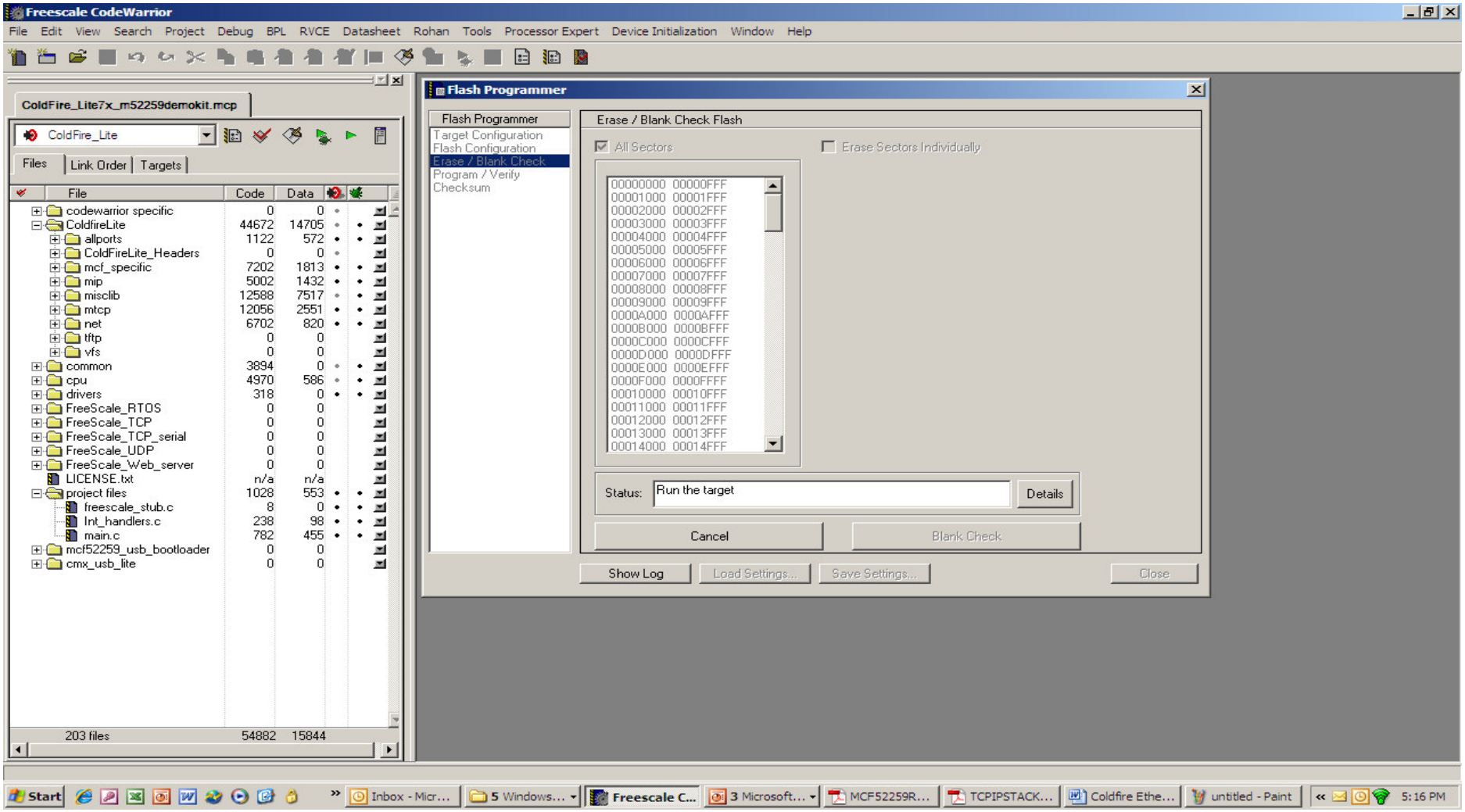
freescale™ semiconductor

Flash file selection



freescale semiconductor

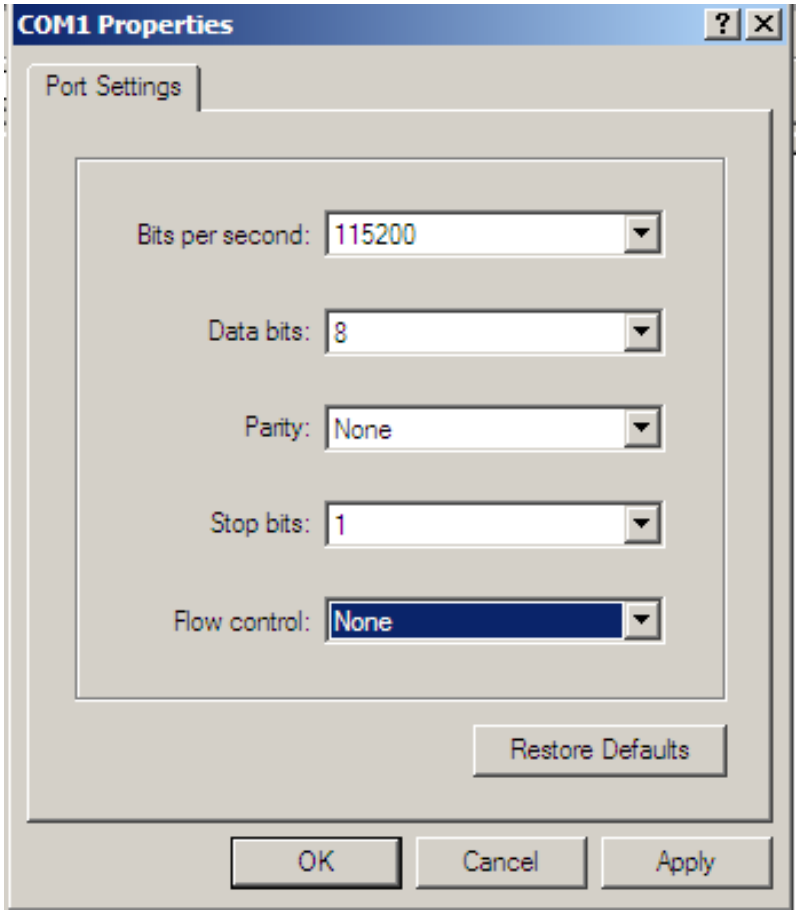
Erase and program the flash



freescale™ semiconductor

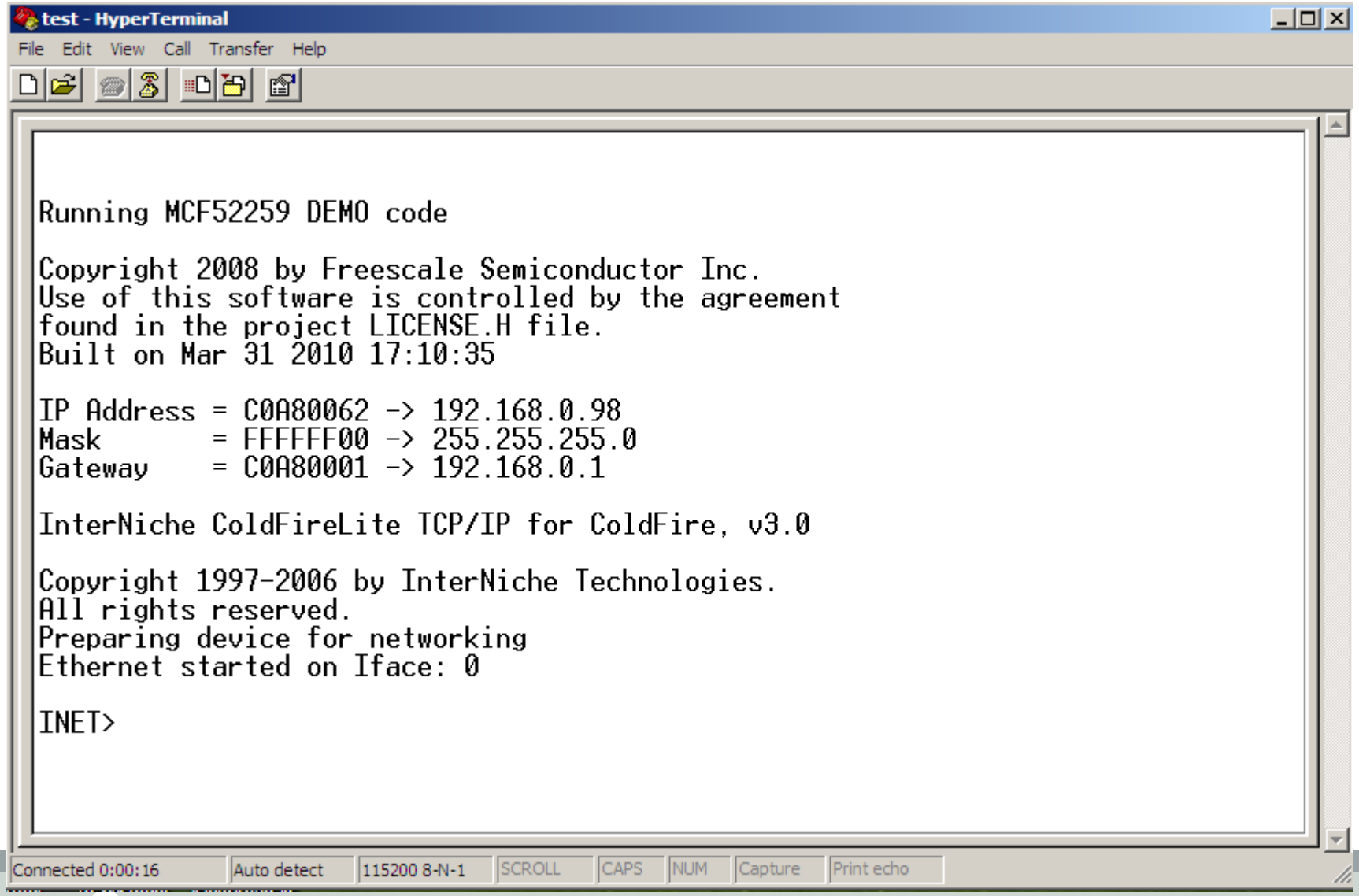
Hyper terminal setting

Baudrate – 115200
Stop bit – NO
Data bit – 8
Parity None



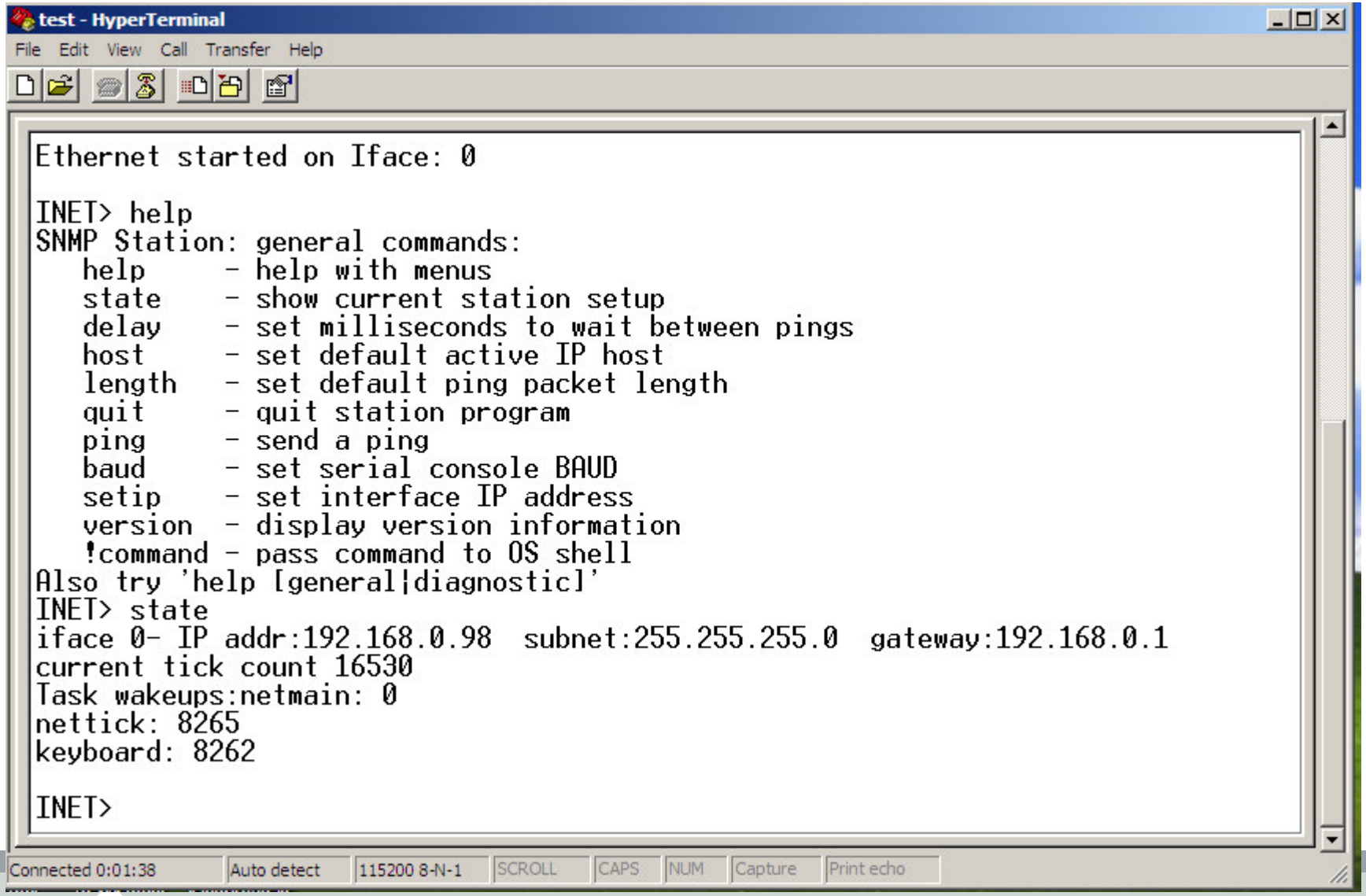
freescale semiconductor

Reset the board- Should receive the message



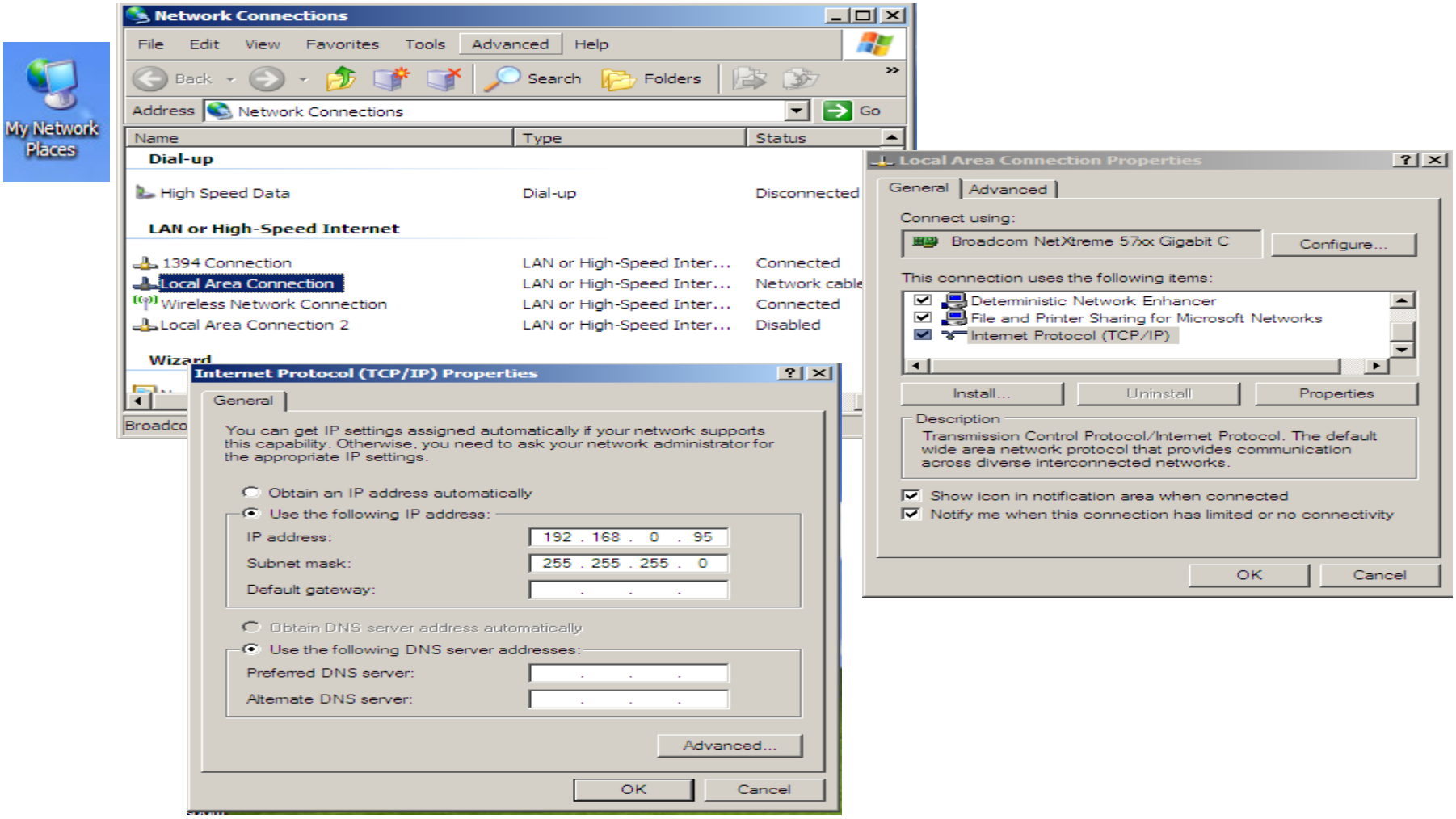
freescale™ semiconductor

Commands



freescale™ semiconductor

Setting IP address at PC side

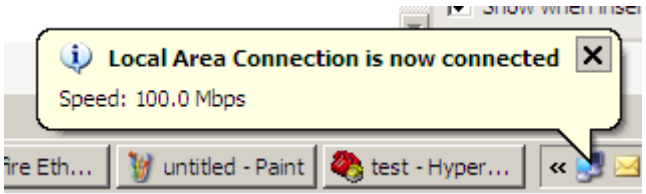


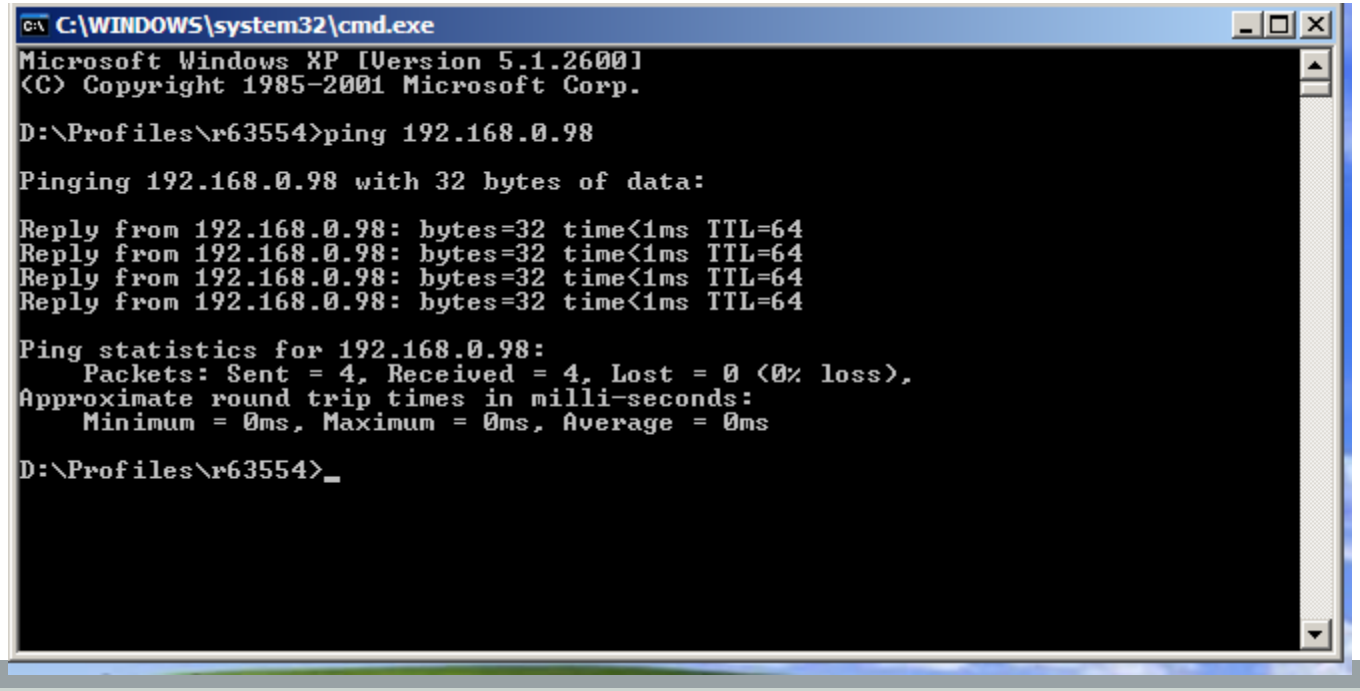
freescale™ semiconductor

Comunicating between the board and PC – PING COMM

Connect the Ethernet cable between the MCF52259 board to PC

Type the command ping “192.168.0.98” from PC side.



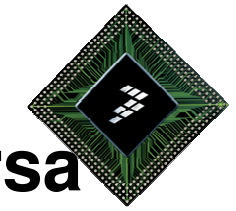


freescale semiconductor

IP Address changing

Open the Main.c file
Change the IP address.
Check with the ping command as well as in boot mointor about the changes you have done for IP address is working or not.

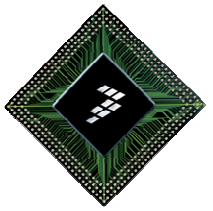
LAB2 – Serial to Ethernet and Visa versa



freescale™
semiconductor

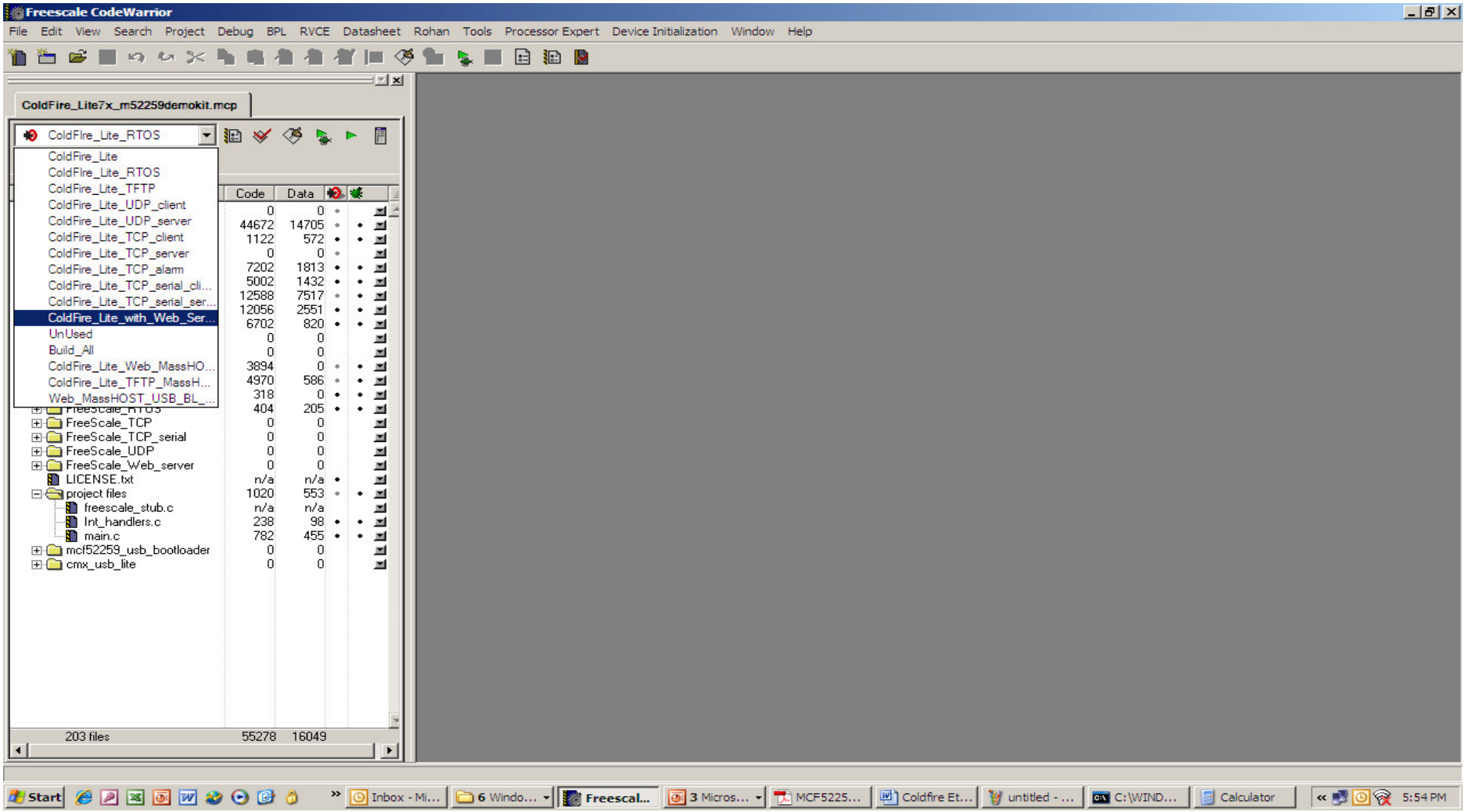
freescale™
semiconductor

LAB3 – Web server



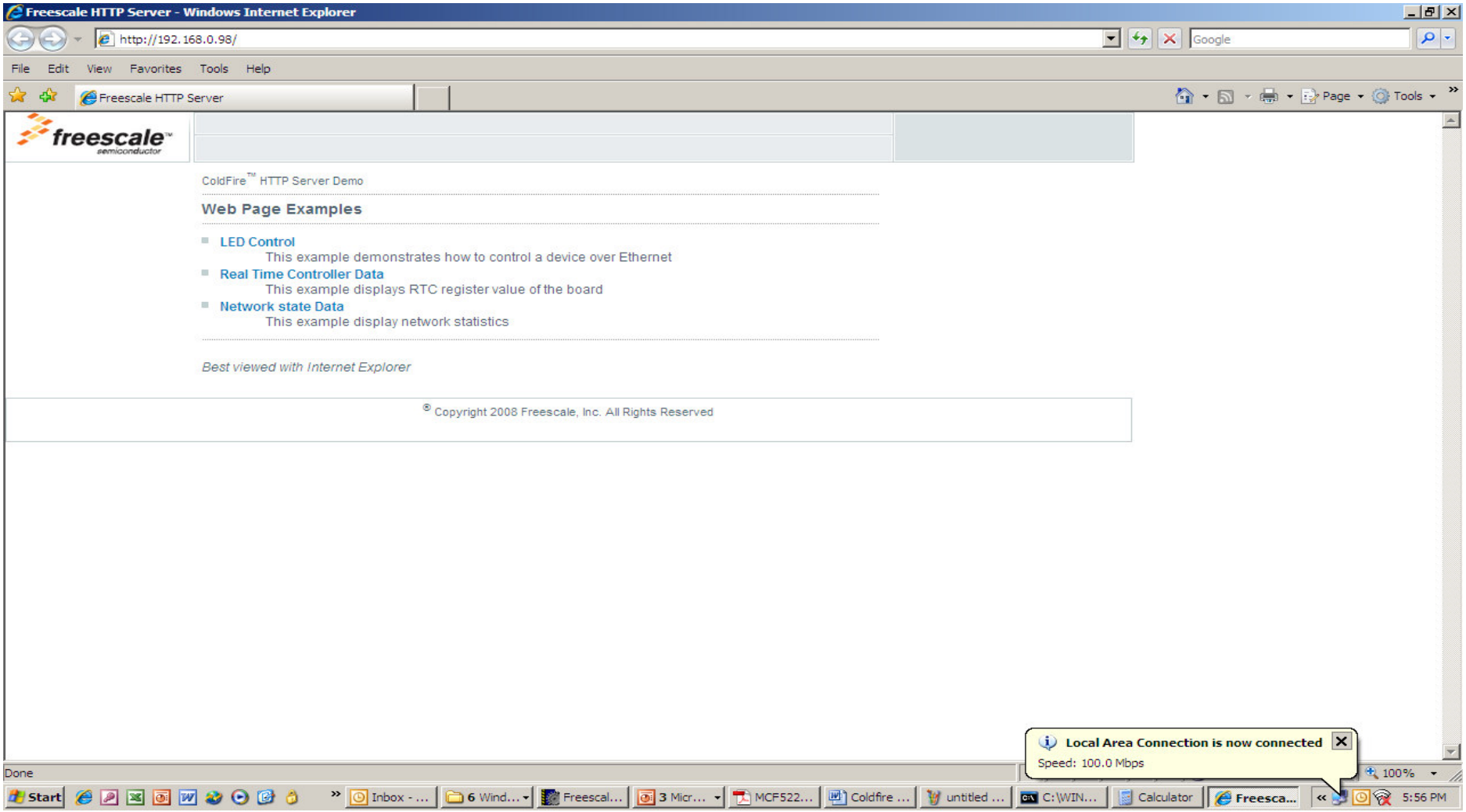
freescale™
semiconductor

Webserver project



freescale™ semiconductor

Webserver demo



freescale semiconductor

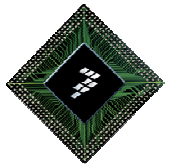

freescale™
semiconductor

TWR-MCF52259-USB

April 7, 2010

Hareesh S

Sr.FAE



K2u/USB Workshops-on-Demand Series

Module Agenda

This module contains:

- Introduction to USB
- Basic Operation of USB hardware and software
- USB Data Structures for K2/3u
- USB API Calls for K2/3u

Introduction To USB

freescale
semiconductor

In This Section:

- Motivation for the USB Standard
- History and Evolution of USB
- USB Topology
- USB Connectors

Motivation for USB

USB - Universal Serial Bus was born out of the need to provide designers and end-users with:

- an alternative to Apple's 1394 digital link standard (FireWire®)
- a fast, bi-directional, low-cost, dynamically attachable serial interface
- that removes the port availability constraints for the PC and other devices

Motivation for USB

freescale™ semiconductor

- and adds true plug-and-play attributes for a wide range of devices simultaneously
- with minimal or no user intervention required for configuration
- and is very end-user friendly

History and Evolution

freescale™ semiconductor

History and Evolution

Current USB Speeds and Limitations

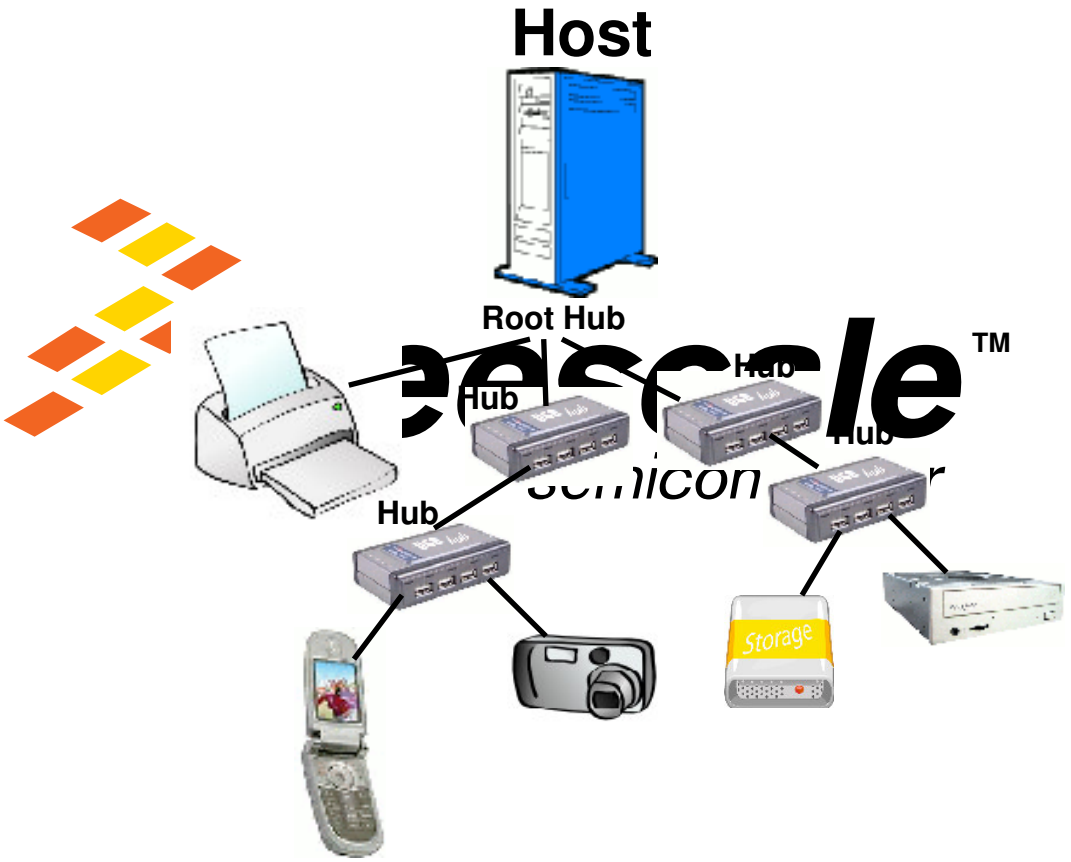
Spec	Data Rate and Performance	Applications	Notes
USB 1.1 Low-speed USB 2.0 Low-speed	1.5 Mbps / 10-[illegible] Kbps	Keyboard, [illegible]	Low cost but limited performance; type and number of endpoints are limited
USB 1.1 Full-speed USB 2.0 Full-speed	12 Mbps / 5-10 Mbps	Printers, audio devices, floppy drives	Moderate performance; guaranteed latency; guaranteed bandwidth
USB 2.0 High-speed	480 Mbps / 25-400 Mbps	Video, storage, imaging	Vast bandwidth improvements

Topology

USB Spec Provides for a Flexible Tiered Star Topology

- Single Host – The host controls communication with each device
- Up to 127 devices can be attached
- Devices are one of the following:
  - Hubs (provide additional attachment points)
  - Functions (devices) which provide capabilities to the system (e.g., printer, thumb drive, I-pod)
- Up to 7 tiers
  - Can have up to 5 hubs deep with a max of 5 meters between each hub

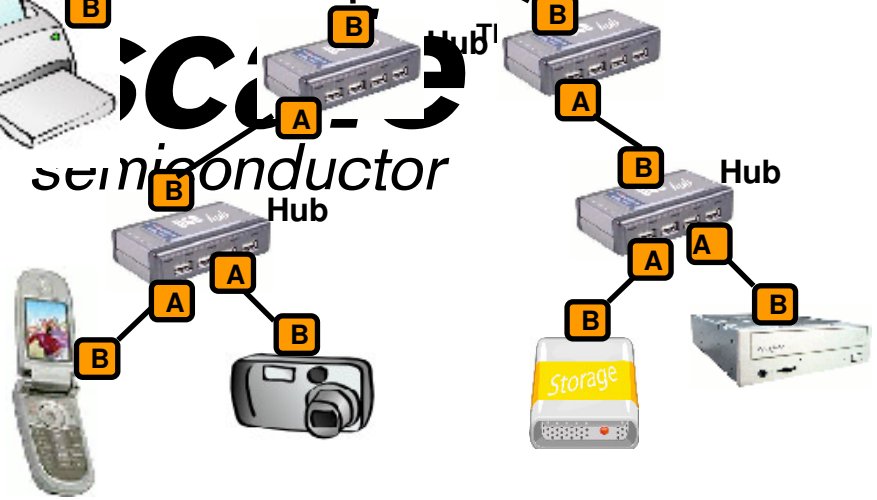
Topology - Tiered Star



USB Standard Connectors

A-connector (upstream)

B-connector (downstream)



Connector types

freescale™
semiconductor

Software General Concepts

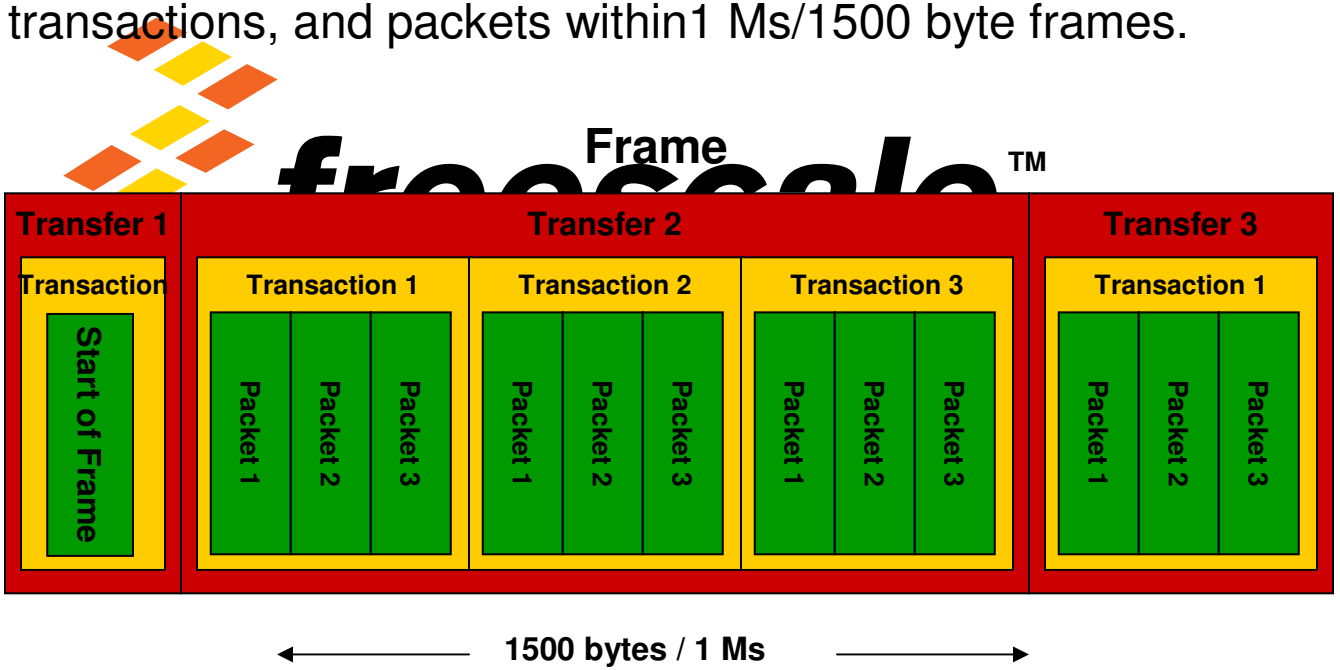
In This Section:

- How Data is Transferred in USB
- How Does the Host Know the Device's Requirements?
- The Enumeration Process: What happens when a device is connected?
- A More Detailed Look at What Comprises a Frame
- API Calls: What happens after enumeration?

The General USB Process

How Data is Transferred in USB

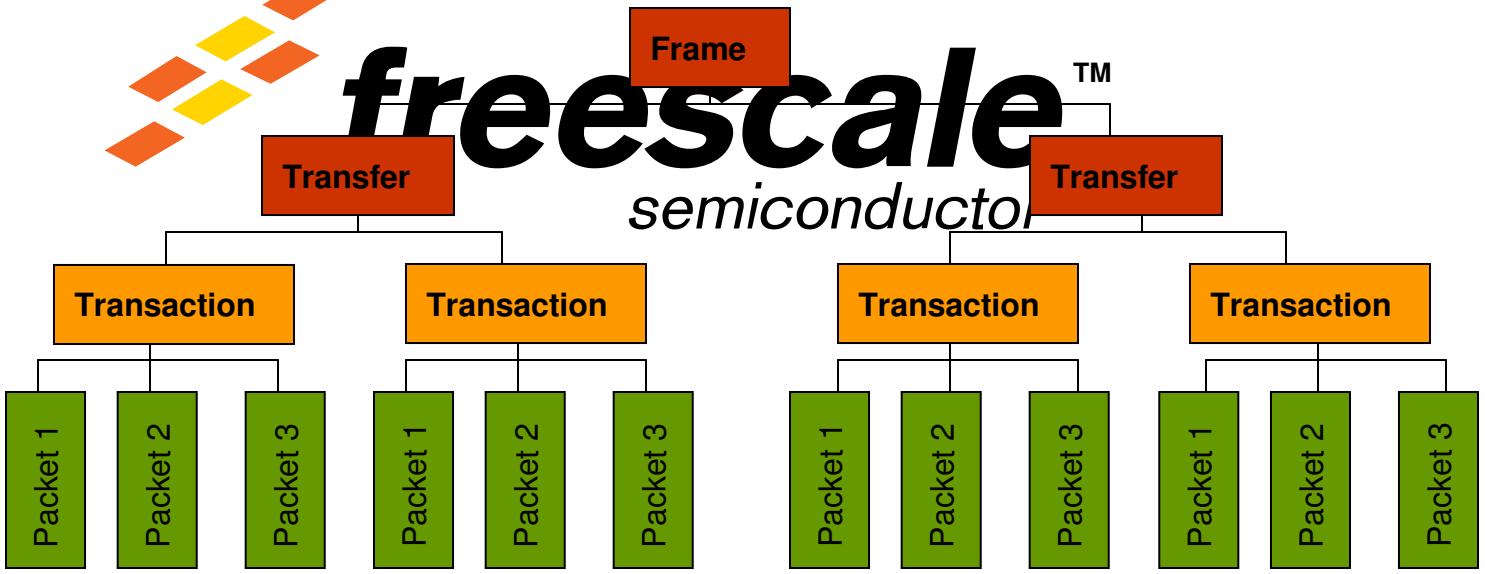
USB is a token-based (packet) standard. Data is transferred between the host and the device in a series of frames, transfers, transactions, and packets within1 Ms/1500 byte frames.



The General USB Process

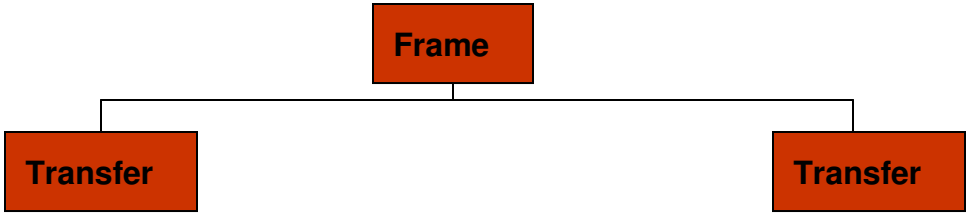
How Data is Transferred in USB

Frames are made up of transfers, which are made up of transactions, which are made up of packets. The USB host schedules these 1mS frames when communicating with the low and full speed devices.



The General USB Process

How Data is Transferred in USB



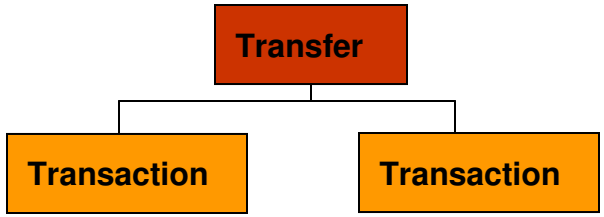
There are 4 types of data transfer:

1. Interrupt
2. Bulk
3. Isochronous
4. Control

The type of transfer depends upon the type of device and its data requirements (this is discussed in more detail later).

The General USB Process

How Data is Transferred in USB

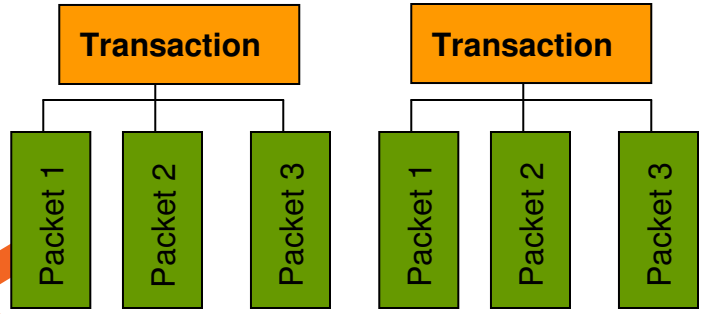


Each transfer can contain multiple transactions and a transaction can span multiple frames.

The host schedules transactions within the 1mS frame.

The General USB Process

How Data is Transferred in USB



For each transaction there are three types of packets that communicate the data between host and device:

1. **Token Packet** – the header that defines what follows
2. **Optional Data Packet** – contains the data being transmitted
3. **Status/Handshake Packet** – used to acknowledge transactions and provide a means of error correction

The General USB Process

How Does the Host Know a Device's Requirements?

When a USB device is plugged into a USB port the host communicates with the device and configures each device according its unique requirements such as:

1. **Type of transfer required (interrupt, bulk, isochronous, or control**
2. **Who supplies the power (host or device)**
3. **Maximum packet size**
4. **The number of configurations (e.g., a single device can be configured to use its own power or power from the host)**
5. **Manufacturer's product ID and registration information**
6. **Etc.**

This configuration process is called enumeration. Enumeration is be covered in greater detail later in this presentation

The General USB Process

How Does the Host Know a Device's Requirements?

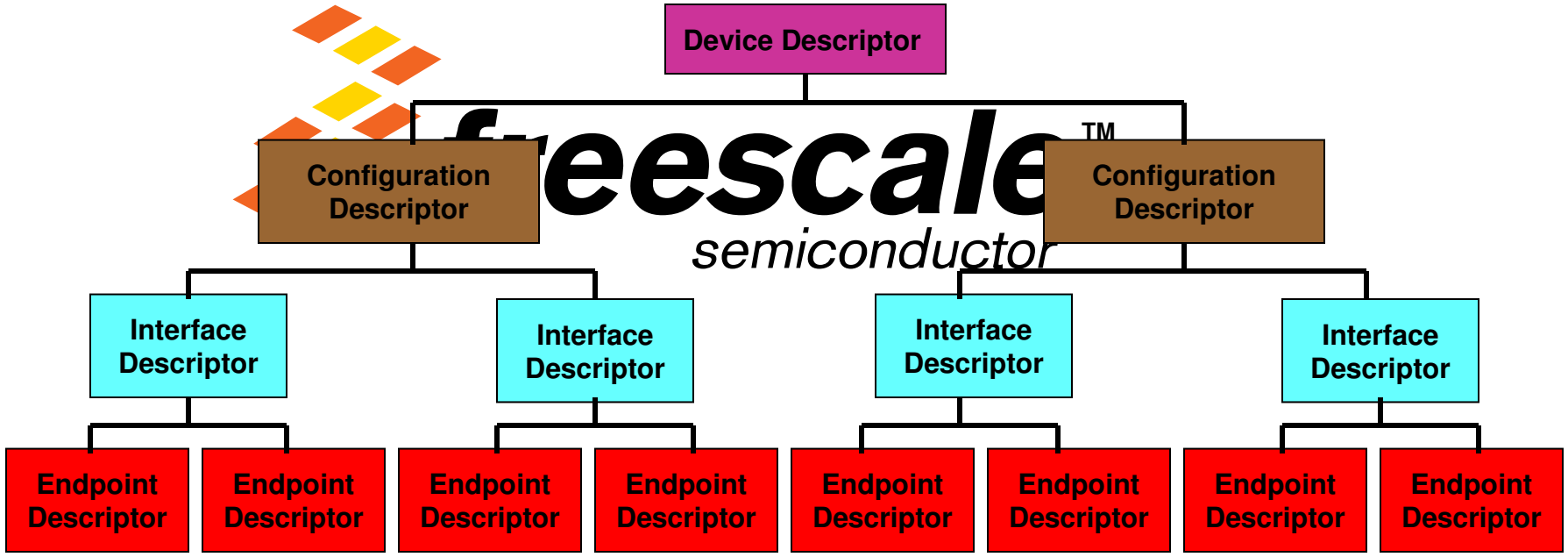
These requirements are communicated to the host through a hierarchy of C program descriptors. The types of descriptors include:

1. **Device Descriptors**
2. **Configuration Descriptors**
3. **Interface Descriptors**
4. **Endpoint Descriptors**
5. **String Descriptors**

The General USB Process

How Does the Host Know a Device's Requirements?

This hierarchy of the most commonly used descriptors looks like this:



The General USB Process

How Does the Host Know a Device's Requirements?

Device Descriptor

Since the device descriptor represents the entire device, there can be only one per device. This descriptor includes basic information such as: USB version supported, maximum packet size, the number of configurations, and vendor and product ID info.

```
//-------------------------------------------------------
// Sample Standard Device Descriptor Type
// Definition Fields
//-------------------------------------------------------
```

Length (18 bytes)
Descriptor Type (DEVICE)
USB Spec Release Number (0200h)
Device class (hub type…Human Interface defined in other descriptor, CDC described here)
Device Sub-class (00h)
Device protocol (00h)
Maximum Packet size (64 bytes – max for the endpoint)
Vendor ID (ID assigned by USB IF)
Product ID (ID assigned by product manufacturer)
Device release number (revision code of device)
Manufacturer (ABC Corp)
Product (string identifier)
Serial Number (1234)
Number of configurations (1 or more configurations can follow)

The General USB Process

How Does the Host Know a Device's Requirements?

Configuration Descriptor

The configuration descriptor is a header to the interface descriptors. It specifies how this configuration of the device is powered, what the maximum power consumption is, and how many interfaces there are. There can be more than one configuration descriptor (for example: if the device can switch between self-power and host-power).

```
//-----------------------------------------------------------
// Sample Standard Configuration Descriptor Type
// Definition Fields
//-----------------------------------------------------------
```

Length (9 bytes)
Descriptor Type (CONFIGURATION)
Total Length (total length in bytes of data returned)
Number of Interfaces (number of interfaces present for this configuration)
Configuration Value (value used by the SetConfiguration request to select this configuration)
Configuration (Index of String Descriptor describing this configuration)
Attributes (bus powered, self powered, remote wakeup)
Max Power (Maximum Power Consumption in 2mA units)

The General USB Process

How Does the Host Know a Device's Requirements?

Interface Descriptor

The interface descriptor groups endpoints into functional groups that perform a single feature of the device.

```
//------------------------------------------------------------
// Sample Standard Interface Descriptor Type
// Definition Fields
//------------------------------------------------------------
```

Length (9 bytes)
Descriptor Type (INTERFACE)
Interface Number (interface number, zero based, and incremented once for each new interface descriptor.)
Alternate Setting (value used to select alternative setting)
Number of Endpoints (number of Endpoints used for this interface)
Interface Class (e.g., HID, mass storage))
Interface Sub Class (subclass Code - assigned by USB Org)
Interface Protocol (Protocol Code assigned by USB Org)
Index (Index of String Descriptor Describing this interface)

The General USB Process

How Does the Host Know a Device's Requirements?

Endpoint Descriptor

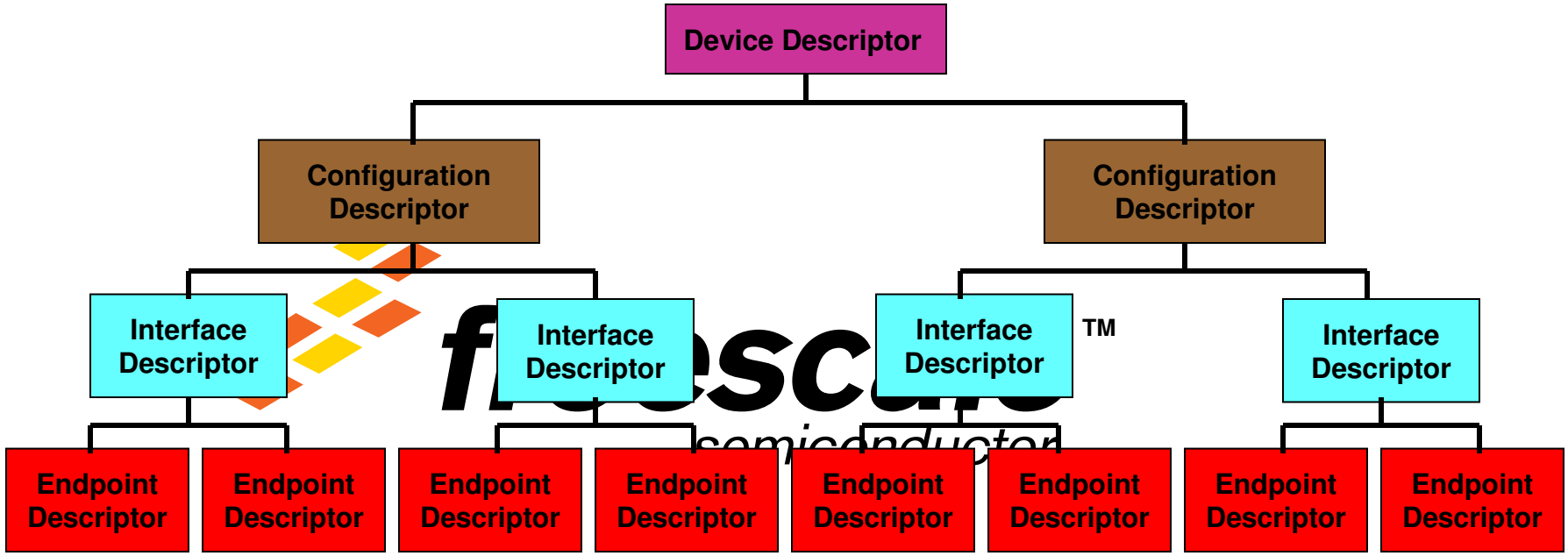
The endpoint descriptors define what transfer type to use, the maximum packet size and time interval used to poll the frames (1mS for low or full speed devices and 125us for high speed devices). From this the host determines bandwidth requirements.

```
//------------------------------------------------------------
// Sample Standard Endpoint Descriptor Type
// Definition Fields
//------------------------------------------------------------
```

Length (7 bytes)
Descriptor Type (ENDPOINT)
End Point Address (indicates what endpoint this descriptor is describing)
Attributes (specifies the transfer type)
Maximum Packet Size (indicates the maximum payload size for this endpoint)
Interval (interval for polling endpoint data transfers. Value in frame counts. Ignored for Bulk & Control Endpoints. Isochronous must equal 1 and field may range from 1 to 255 for interrupt endpoints)

The General USB Process

How Does the Host Know a Device's Requirements?

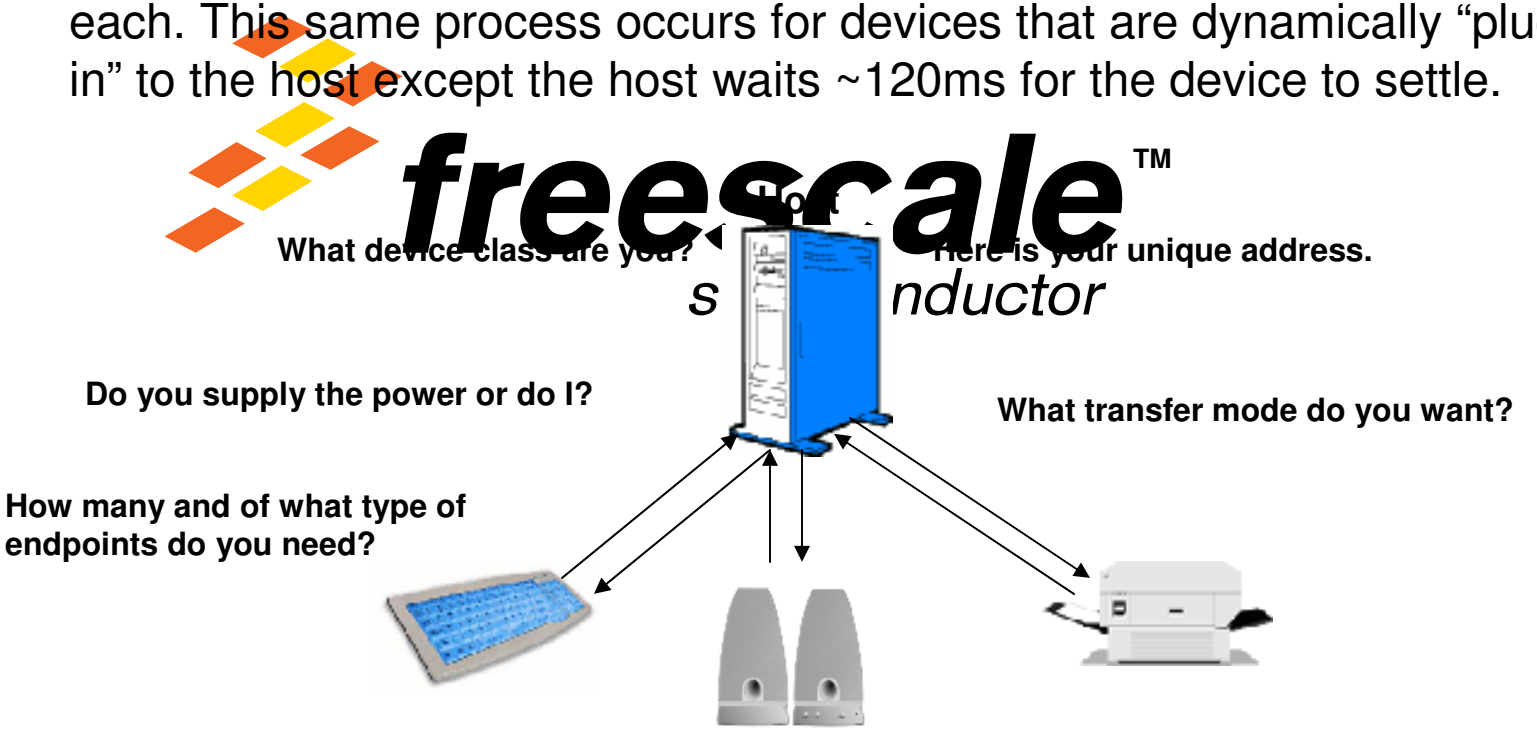


Upon detecting a USB device connection the host, using these descriptors, configures the device and loads the proper driver program. This configuration process is called "**enumeration**."

The General USB Process

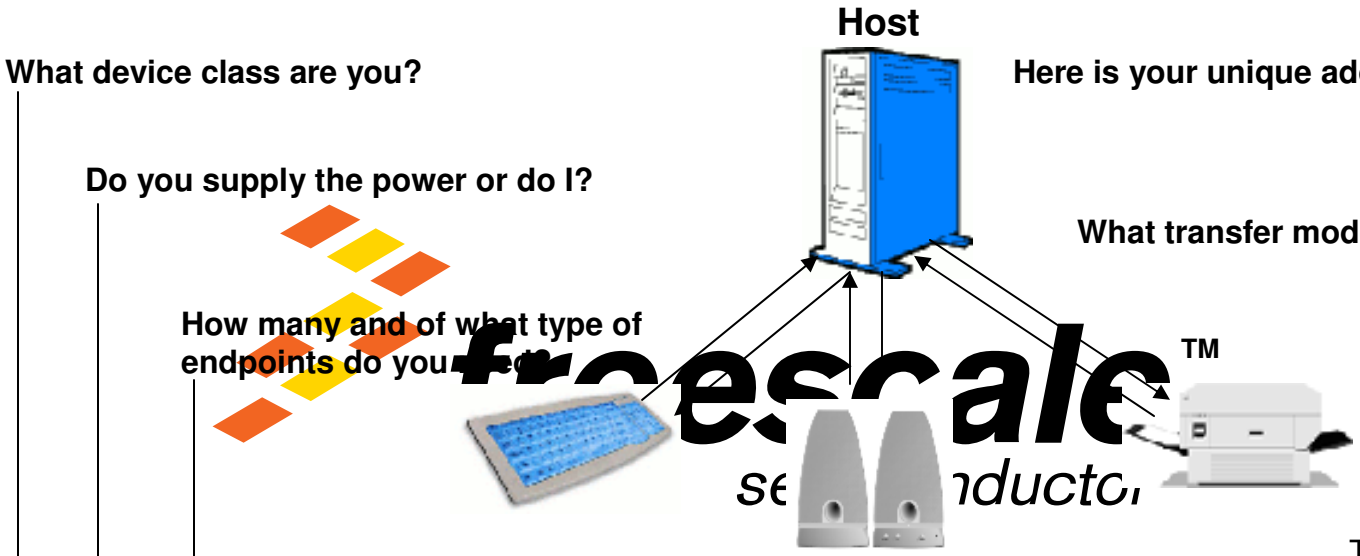
The Enumeration Process: What happens when a device is connected?

1. **Enumeration** -Upon powering up, the host: queries all connected devices to determine the requirements of each (such as, class of device power source, number and types of endpoints) and assigns a unique address for each. This same process occurs for devices that are dynamically "plugged in" to the host except the host waits ~120ms for the device to settle.



The General USB Process

The Enumeration Process: What happens when a device is connected?



Endpoints: A uniquely identifiable portion of a Universal Serial Bus device that is the source or sink of information in a communication flow between the host and device.

Power can be supplied by the host or the device.

USB Classes have been defined that include class-specific descriptors (e.g. HID class and mass-storage class)

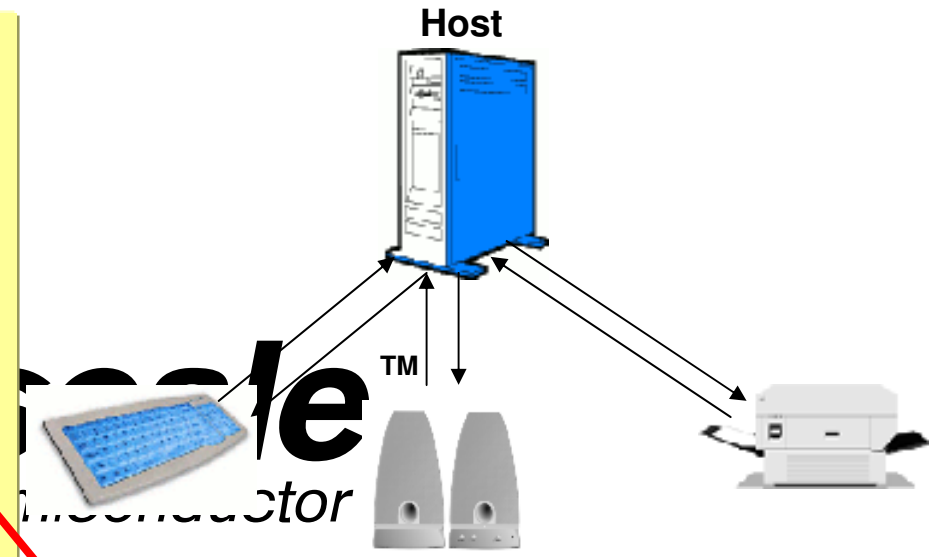
There are 4 types of data transfer:

1. Interrupt
2. Bulk
3. Isochronous
4. Control

The General USB Process

The Enumeration Process: What happens when a device is connected?

//----------
// Sample Standard Device Descriptor Type
// Definition Fields
//----------
Length (18)
Descriptor Type (DEVICE, CONFIGURATION, INTERFACE, ENDPOINT, HID)
USB Spec Release Number (0200h)
Device class (hub type…Human Interface defined in other descriptor, CDC described here)
Device Sub-class (00h)
Device protocol (00h)
Maximum Packet size (64 bytes – max for the endpoint)
Vendor ID (ID assigned by USB IF)
Product ID (ID assigned by product manufacturer)
Device release number (revision code of device)
Manufacturer (ABC Corp)
Product (string identifier)
Serial Number (1234)
Number of configurations (1 or more configurations can follow)



Here is your unique address.

What device class are you?

What transfer mode do you want?

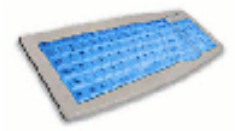
Do you supply the power or do I?

How many and of what type of endpoints do you need?

The General USB Process

The Enumeration Process: What happens when a device is connected?

2. **Transfer Mode** - The host then determines which type of transfer each device requires. There are three transfer modes:



Interrupt Transfer - Devices that send very little data such as mice or keyboards would choose this type of transfer. 64 Kbytes/second transfer rate

Bulk Transfer - Devices that send or receive large packets that need to be verified as accurate, such as printers, would choose this type of transfer. 1216 Kbytes/second transfer rate

Isochronous Transfer- Devices that stream data such as speakers would choose the Isochronous type of transfer. There is no error correction with this transfer mode as with bulk. 1023 Kbytes/second transfer rate

Control Transfer - The host sends commands, and query parameters via control packets. All devices use Control transfers to Endpoint 0 for the Enumeration process. 832 Kbytes/second transfer rate

The General USB Process

The Enumeration Process: What happens when a device is connected?

Interrupt transfer mode

isochronous transfer mode

bulk transfer mode

3. As the host **enumerates** each device it sets aside bandwidth for the devices that use the **interrupt** and **isochronous transfer mode**s and keeps track of the total bandwidth used. These two transfer modes can use up to 90% of the total bandwidth.

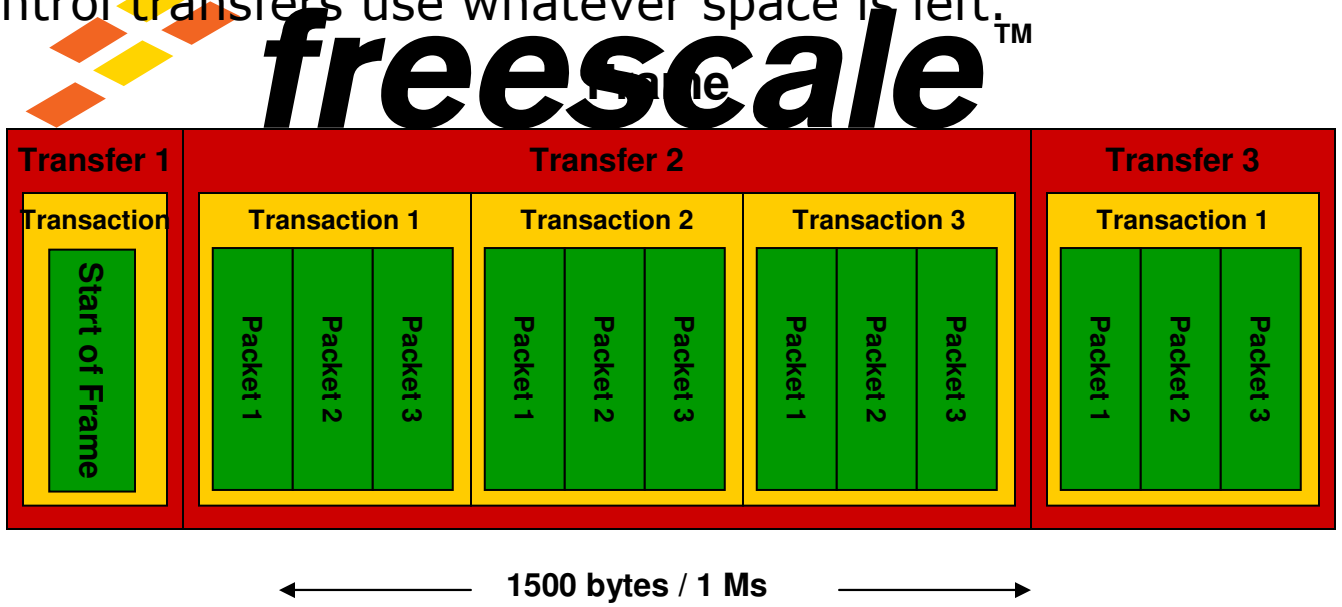
freescale™ semiconductor

4. Once 90% of the available bandwidth is used up, the host does not allow any other **interrupt** or **isochronous** devices to be **enumerated** and the host uses the remaining bandwidth of at least 10% for **control transfer packets and bulk transfer packets**.

The General USB Process

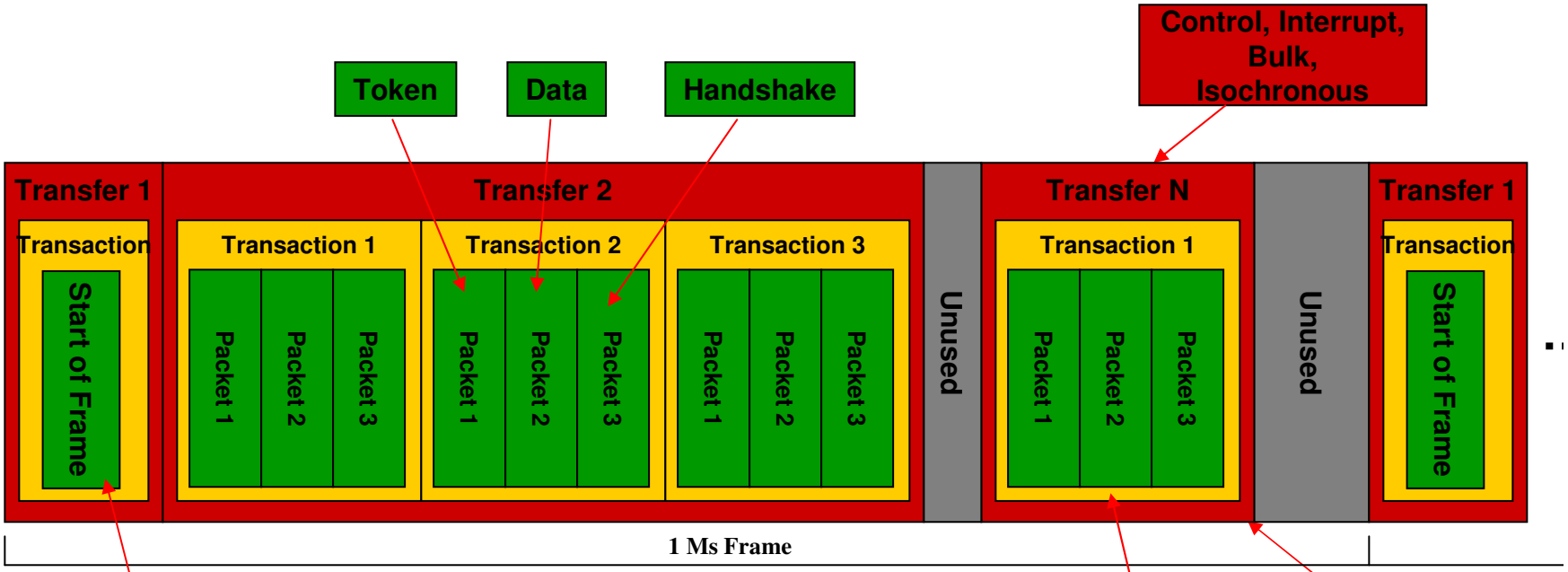
The Enumeration Process: What happens when a device is connected?

5. The available bandwidth is then divided into **frames**, and the host controls those frames which contain 1,500 bytes. Every millisecond a new frame begins. Within the frame, slots are reserved for isochronous and interrupt devices (up to 90%) so that they are guaranteed the bandwidth they need. Bulk and control transfers use whatever space is left.



The General USB Process

Frames in More Detail



Each frame begins with a Start of Frame packet followed by the hosts transactions to device endpoints for data transfer.

The host schedules transactions within 1mS frames for low and full speed.

The host may schedule transactions anywhere in the frame, but they must complete within the frame.

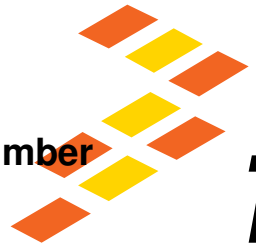
Transfers may span multiple frames.

The General USB Process

Packets in More Detail

Packets are a block of information with a defined data structure. The packet is the lowest level of the USB transfer hierarchy describing the physical layer of the interface. If you were to monitor D+ and D- you would see the packet fields:

Packet Identifier
Address
Endpoint
Data
Frame number
CRC



Token Packet format:

Field	PID	Address	Endpoint	CRC
Bits	8	7	4	5

Start of Frame Packet format:

Field	PID	Frame Number	CRC
Bits	8	11	5

freescale semiconductor

Data Packet format:

Field	PID	Data	CRC
Bits	8	0-1023	16

Handshake Packet format:

Field	PID
Bits	8

CRC covers everything in the packet with the exception of the PID which has its own error checking mechanism.

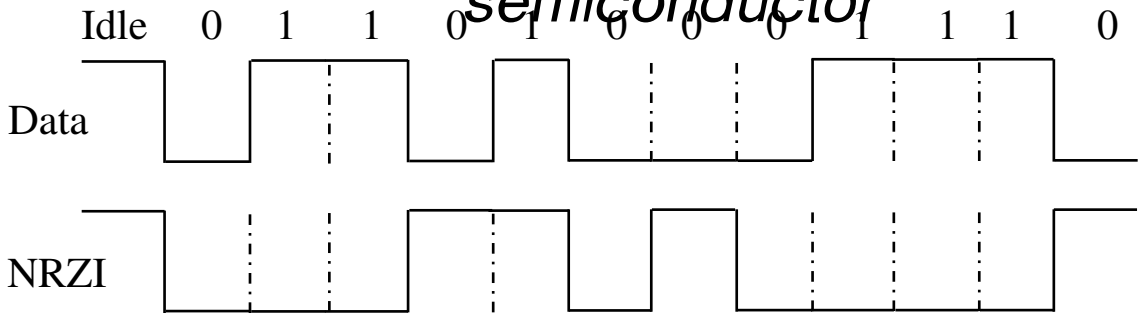
The General USB Process

Packets Associated with Each Type of Transfer

Transfer Type	Stages (Transactions)	Phases (Packets)
Control	Setup	Token
		Data
		Handshake
	Data (IN or OUT) (optional)	Token
		Data
		Handshake
	Status (IN or OUT)	Token
		Data
		Handshake
Bulk	Data (IN or OUT)	Token
		Data
		Handshake
Interrupt	Data (IN or OUT)	Token
		Data
		Handshake
Isochronous	Data (IN or OUT)	Token
		Data

USB Signaling

- USB uses Non-Return to Zero, Inverted signaling
  - Separate clock signal not required to be delivered with data
  - '1' on the data line indicates no change in level of NRZI signal
  - '0' on the data line indicates transition in level of NRZI signal
  - Long string of 1s will eventually cause receiver to lose synchronization



API Calls: What happens after enumeration?

After enumeration the device is ready to perform the function it was designed for. It does so by issuing a set of properly sequenced API calls.

Definition:

Application Program Interface (API): A formalized set of firmware calls and routines that can be referenced by an application program to access driver functions.

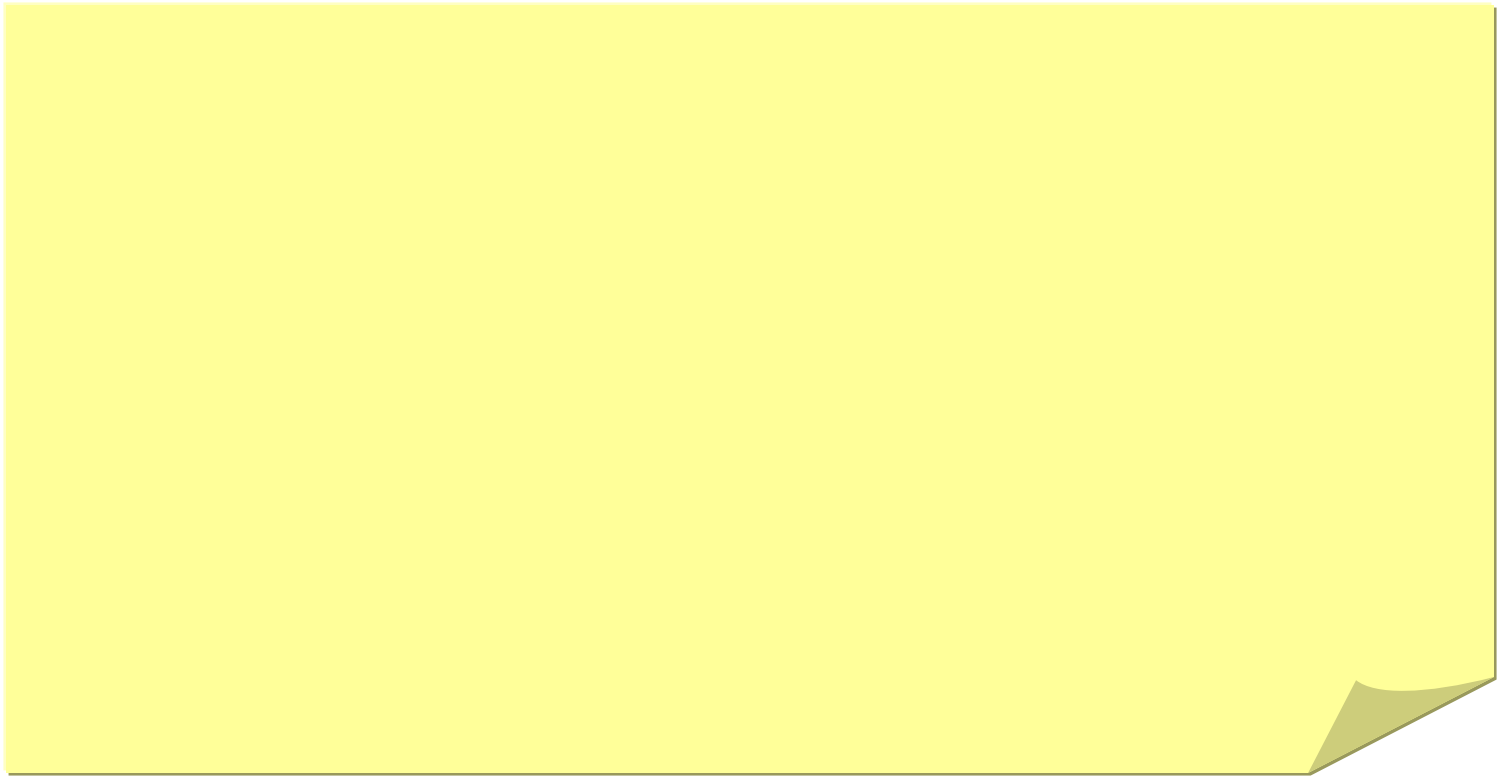
API Calls: What happens after enumeration?

Commonly used API calls for host devices:

•host_init()	- Initialize the host controller
•host_stop()	- Disable the host controller
•host_reset_bus()	- Reset the USB bus and all connected devices
•host_send_control()	- Send configuration info
•host_receive_control()	- Get config info
•host_send()	- Send data on USB channel
•host_receive()	- Get data on USB channel
•host_add_ep()	- Define an EP for your use
•host_remove_ep()	- Delete an EP (allows others to use it)
•host_modify_ep()	- host_modify_ep()
•host_ms_delay()	- host_ms_delay()
•host_scan_for_device()	- Application scan USB see devices attached

API Calls: What happens after enumeration?

Commonly used API calls for devices:



API Calls: What happens after enumeration?

Note to reviewers…. what happens behind the scenes (after enumeration) between the host and device driver etc. (e.g., the host automatically polling the devices for data, receiving the data, and returning the status code, etc.) …….needs to go here.

semiconductor

Note to reviewers….A short introduction to the demos should go here.

Note to reviewers….Here should be a short discussion of what the programmer would have to do to modify the code in the Demo Kit or CMX Offerings for their specific device….e.g. they may just need to modify some descriptors and API calls.

semiconductor

CMX Product Offering - Complimentary Software

Freescale and CMX have collaborated to provide a complimentary USB stack for ColdFire Microcontrollers

Complimentary SW for the MCF522xx Family of Microcontrollers:

•USB Basic Device Controller

•HID device layer for the USB driver
- Generic HID
- HID keyboard
- HID mouse
- HID joystick

CDC to UART bridge functionality

•Host HID Drivers
- Generic HID
- HID keyboard
- HID mouse
- HID joystick

Host Mass Storage

•OTG Drivers
- Session request protocol
- Host negotiation protocol (HNP)
- Vbus "on the fly" control

CMX Product Offering - Upsell Software

These additional stacks are available for purchase from CMX

Reviewers: Please update the delivery schedules here

USB Device

- Reliable Bootloader
- Embedded Pipe
- Mass Storage (for attached SD card or similar)
- FAT file System (for attached SD card or similar)

USB Host

- FAT file System
- Printer Lite

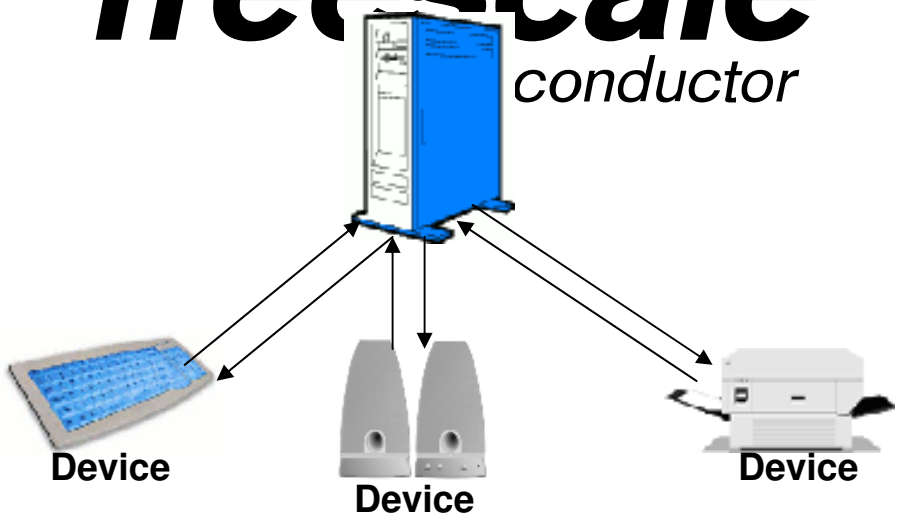
Appendix A

USB On-the-Go

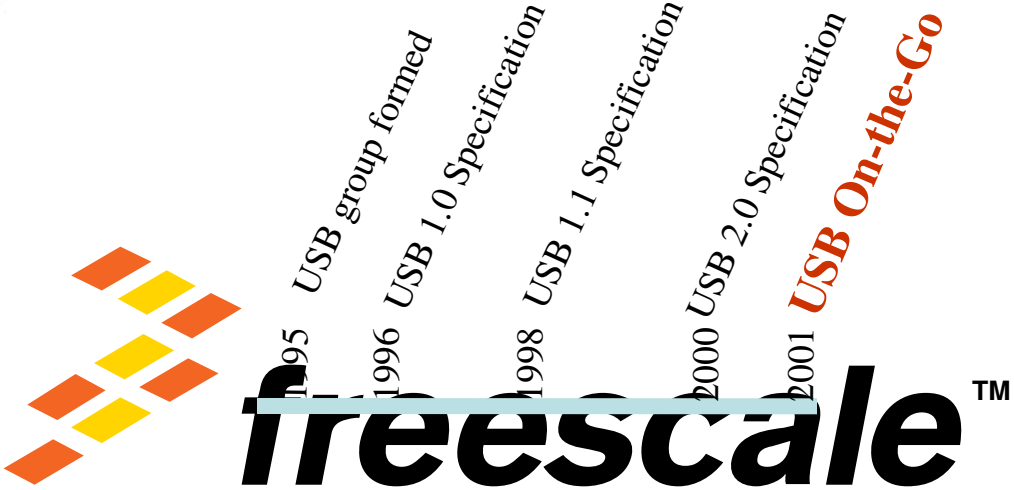
USB On-the-Go: What is it?

Remember that previously we said that USB:

1. can be implemented as a device or a host
2. there can be only one host but many devices
3. and the host controls the devices.

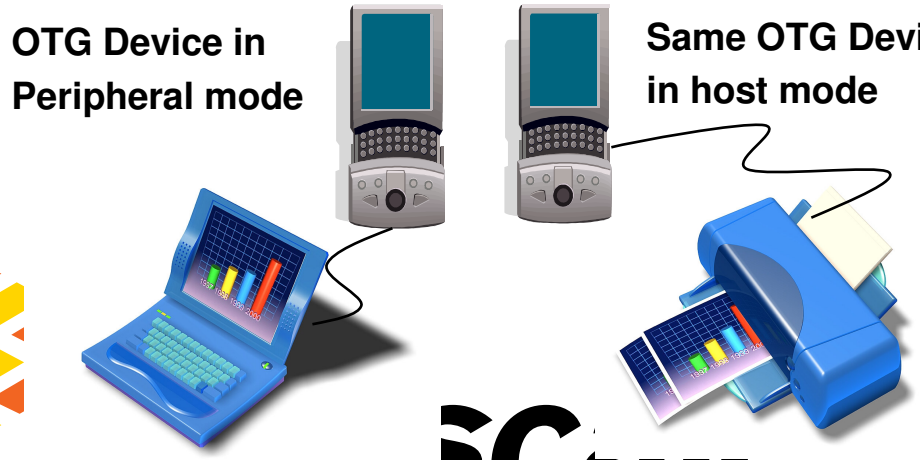


USB On-the-Go: What is it?



By 2001, there was a proliferation of peripheral devices that could be interconnected and pass data to each other. Because of this interaction of peripheral devices, the USB Group saw the need for a USB device to sometimes act as a device and sometimes act as the host. From that need came the USB On-the-Go specification.

On-the-Go: What Applications would use it?



Above is a typical applications that would use USB On-the-Go. On the left laptop is acting as the host controlling the flow of data to the camera phone and on the right the camera phone is acting as the host controlling the flow of data to the printer. Depending upon the application, a device can dynamically decide whether it will be the host or the device. Other possible applications include:


Download Songs


Keyboard Input


Transfer Files


Swap Songs

USB On-the-Go: How does it work?

To implement OTG, two new protocols were added in the USB 2.0 addendum. These new protocols allow USB OTG host wake-up and role reversal respectively. They are:

SRP = Session Negotiation Protocol
HNP = Host Negotiation Protocol

SRP introduces these new/expanded states:

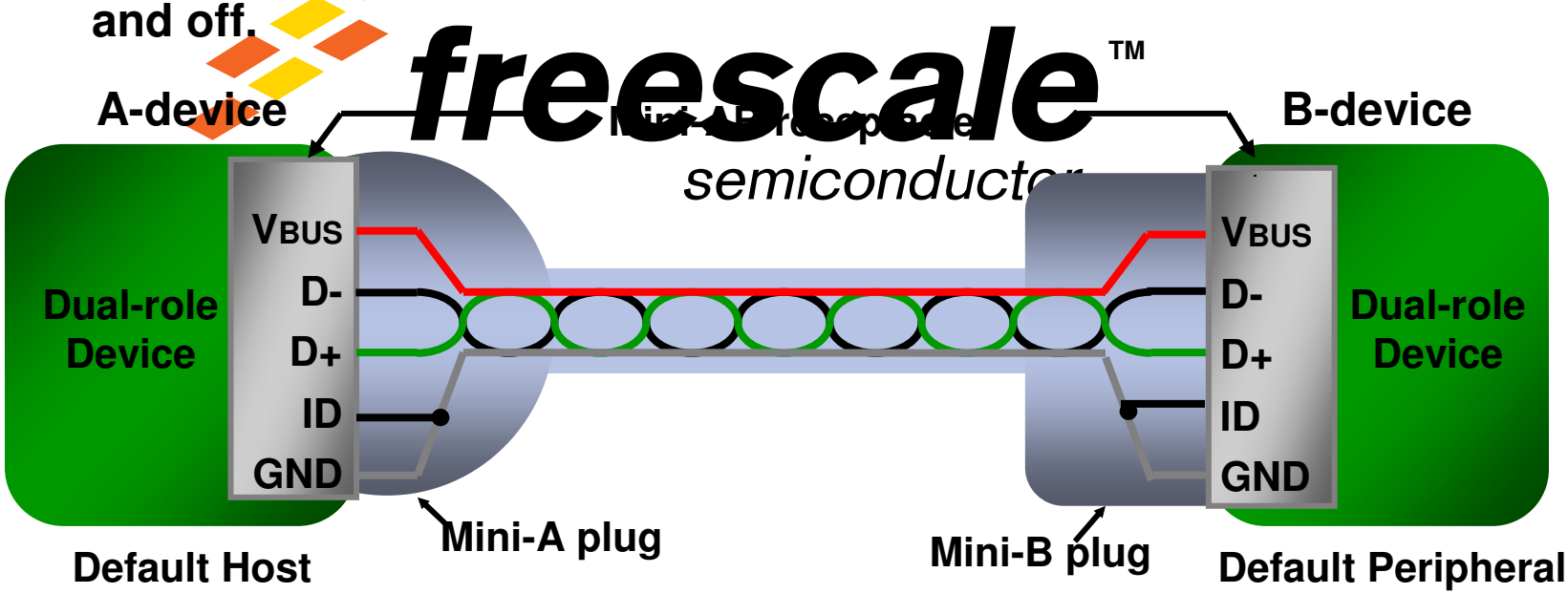
- OTG-A device: a_idle, a_wait_vfall, b_idle
- OTG-B device: b_srp_init

HNP introduces these new/expanded new states:

- OTG-A device: a_peripheral, a_suspend, a_idle
- OTG-B device: b_idle, b_host, b_wait_acon, a_idle

USB On-the-Go: Initial declaration

When using A and B plugs, initially the device using the A plug is declared the host and the one using the B plug is declared the device. When using the mini-AB receptacle the ID plug pin declares the initial state of a device as host or device. State is then changed during host negotiations protocol by turning a pull-up resistor on and off.



USB On-the-Go: Negotiation process

SRP (Session Request Protocol) Negotiations

OTG-B (device) asks OTG-A (host) for a USB OTG session by signaling in one of two ways:

- Pulsing an analog data line (D+ or D-), and
- Pulsing V_{BUS} through a relatively high impedance (> 281 Ω)

freescale semiconductor

Note: The B-device pulses data line first and then pulses V_{BUS}

USB On-the-Go: Negotiation process

HNP (Host Negotiation Protocol):

1. OTG-A (host) enables OTG-B (device) to become host by sending SetFeature (b_hnp_enable) command to OTG-B (device).
2. OTG-A (host) suspends bus signaling so that OTG-B (device) can now become host.
3. OTG-B (device) detects Suspend condition and turns off pull-up resistor.
4. Because HNP is enabled, OTG-A (host) interprets this "disconnect" as a request by the OTG-B (device) to become host.
5. OTG-A (host) turns on its pull-up resistor and becomes peripheral/device.

Continued on the next slide →

USB On-the-Go: Negotiation process

Host Negotiation Protocol continued:

6. To return control back to OTG-A (host) , OTG-B (device) stops using bus and becomes peripheral/device.
7. OTG-A (host) sees lack of activity, disconnects and becomes host.

freescale semiconductor

Note: If the OTG-B (device) does not STALL the SetFeature(b_hnp_enable) command, the OTG-A (host) must give the OTG-B (device) an opportunity to become host before the OTG-A (host) may turn off V_{BUS}.

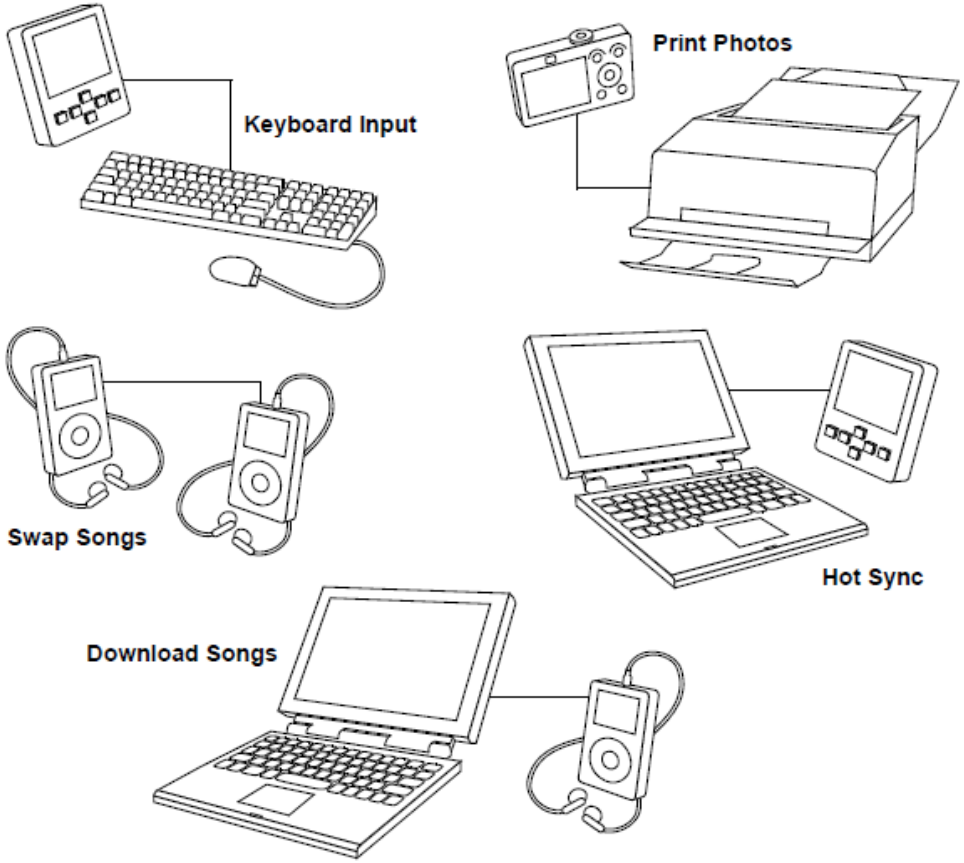


Figure 16-2. Example USB 2.0 On-The-Go Configurations

Freescale USB Solutions

semiconductor

The USB Controller Continuum. Only from Freescale.

8-bit 32-bit

freescale™ semiconductor

- USB is becoming the standard in Consumer and Industrial applications
- Designers want a “one stop shop” USB solution.

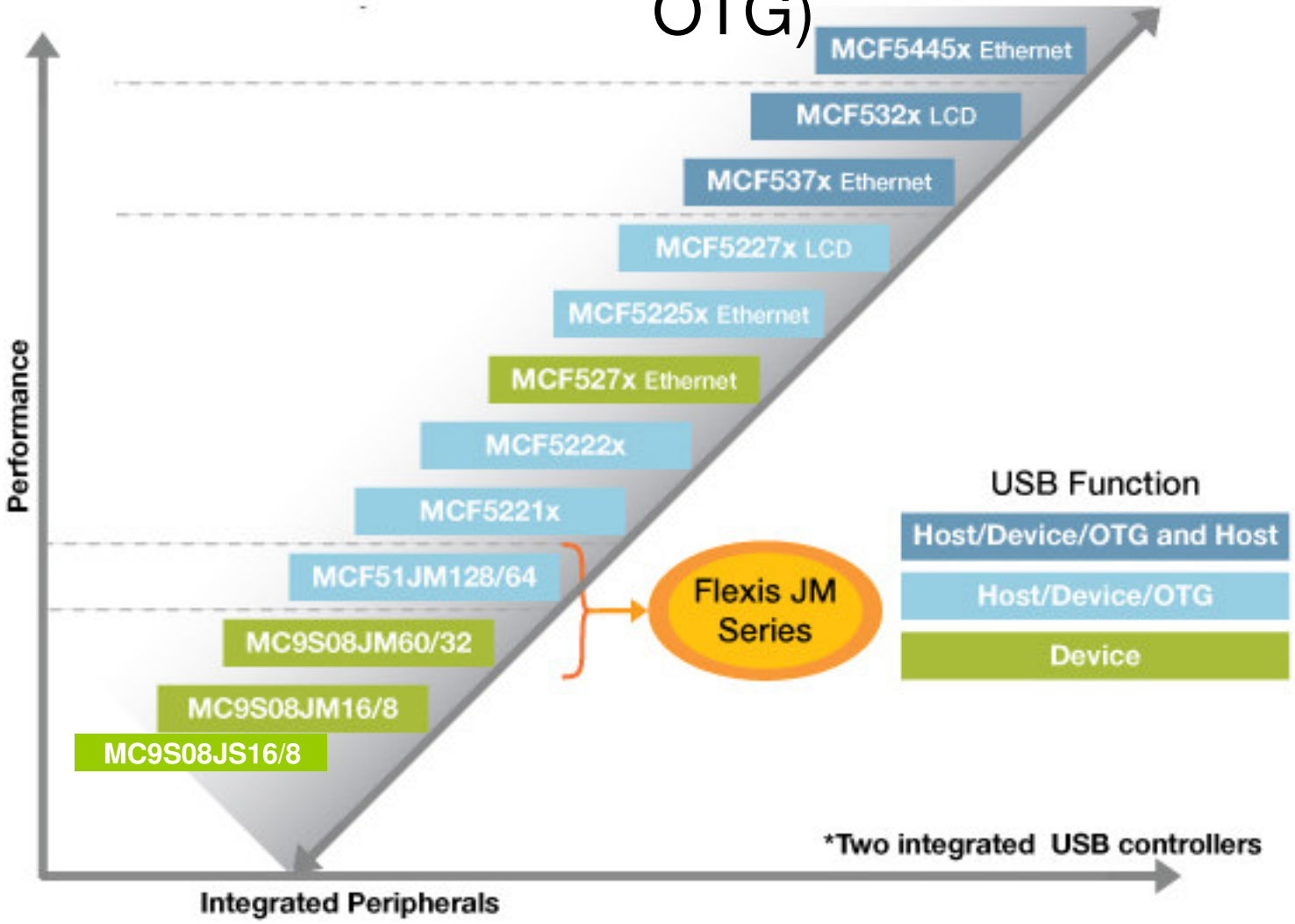
► Freescale provides a complete USB solution to help you enable competitive applications

- Industrial leading wide range portfolio: from cost-effective entry level MC9S08JS8 with 8KB flash and 20 pin packages to powerful MCF5225x with 512KB flash and 144 pin packages
- Market leading software and support: CodeWarrior development tools, Freescale MQX RTOS, USB software stacks, and 3rd party software and tools, training and support from Freescale
- Cost-effective and full featured development tool with getting started DVD help you run up in minutes

"A panel of distinguished judges, representing a cross section of Chinese industry and academic specialists, plus EDN China's editorial board, narrowed the list of award finalists from the total submitted. The award was decided by votes from readers and registered users of EDN China. "

- **EDN China 2008 Innovation Award**
- **winner of the Consumer Electronics IC category**
- **MCF51JM128**

USB Solutions (Low/Full Speed, Device, Host, OTG)



MCF52259 USB features

- Dual Mode USB Controller
- USB 1.1 and 2.0 complient full speed device controller
- 16 bi-directional end points
- DMA or FIFO data stream interface
- Low power consumption
- OTG protocol logic

USB

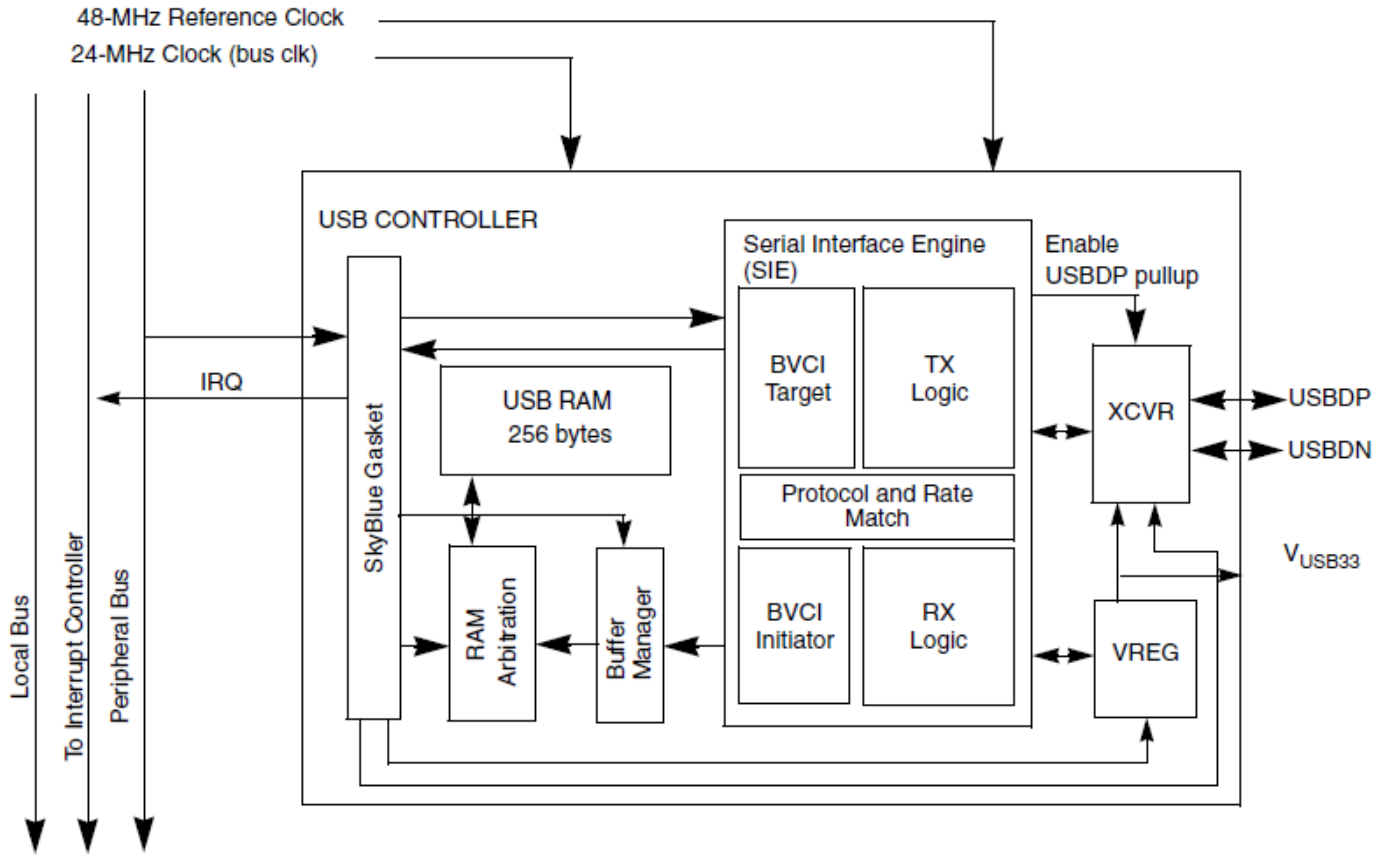


Figure 1. USB Device Controller Block Diagram

LABs

semiconductor

LABs

- LAB1 – HID (Device operation)
- LAB2 – CDC (Device Operation)

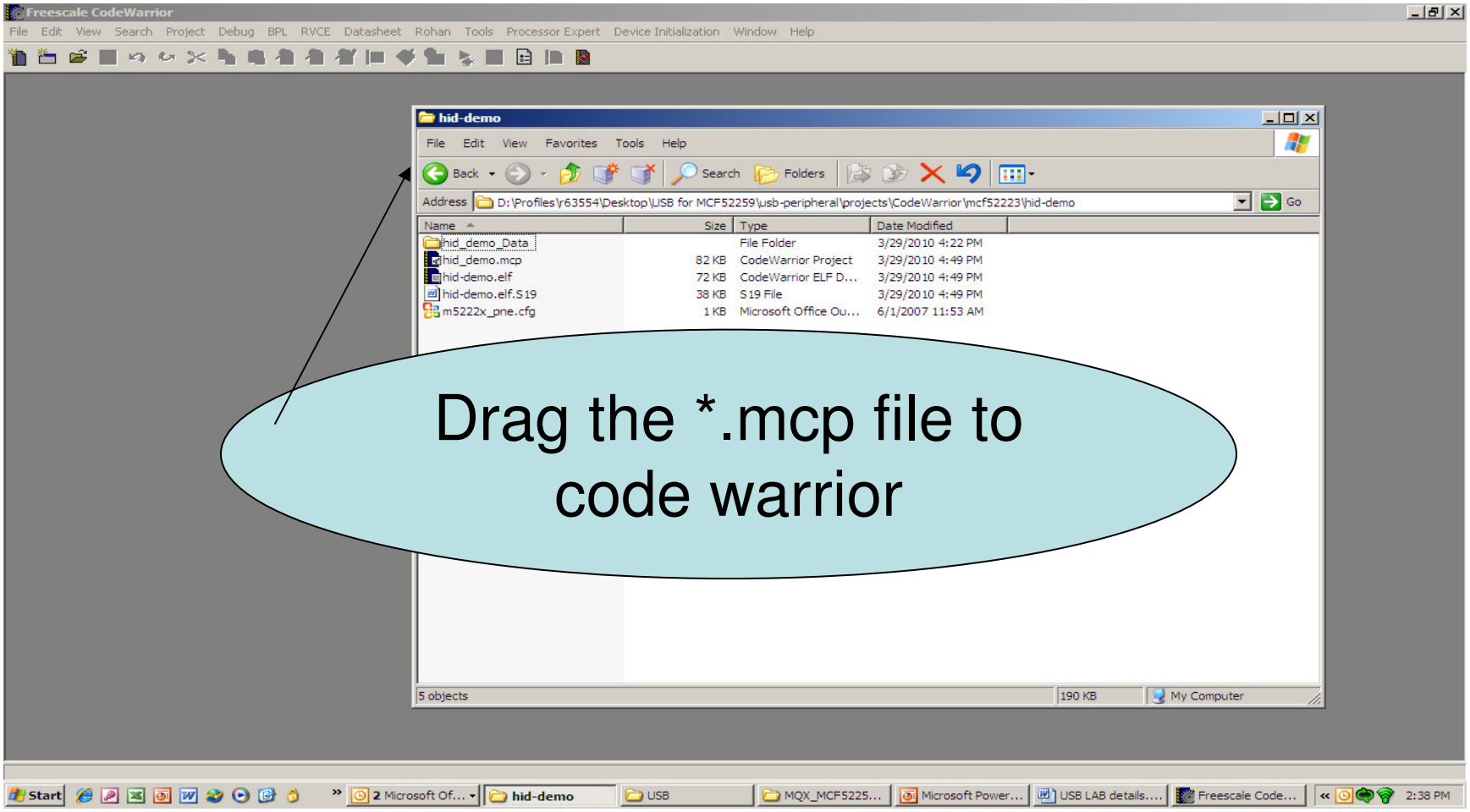
- LAB3 – MSD(Host Operation)
- LAB4 – HID (Host Operation)

LAB1 – HID (Human interface device)

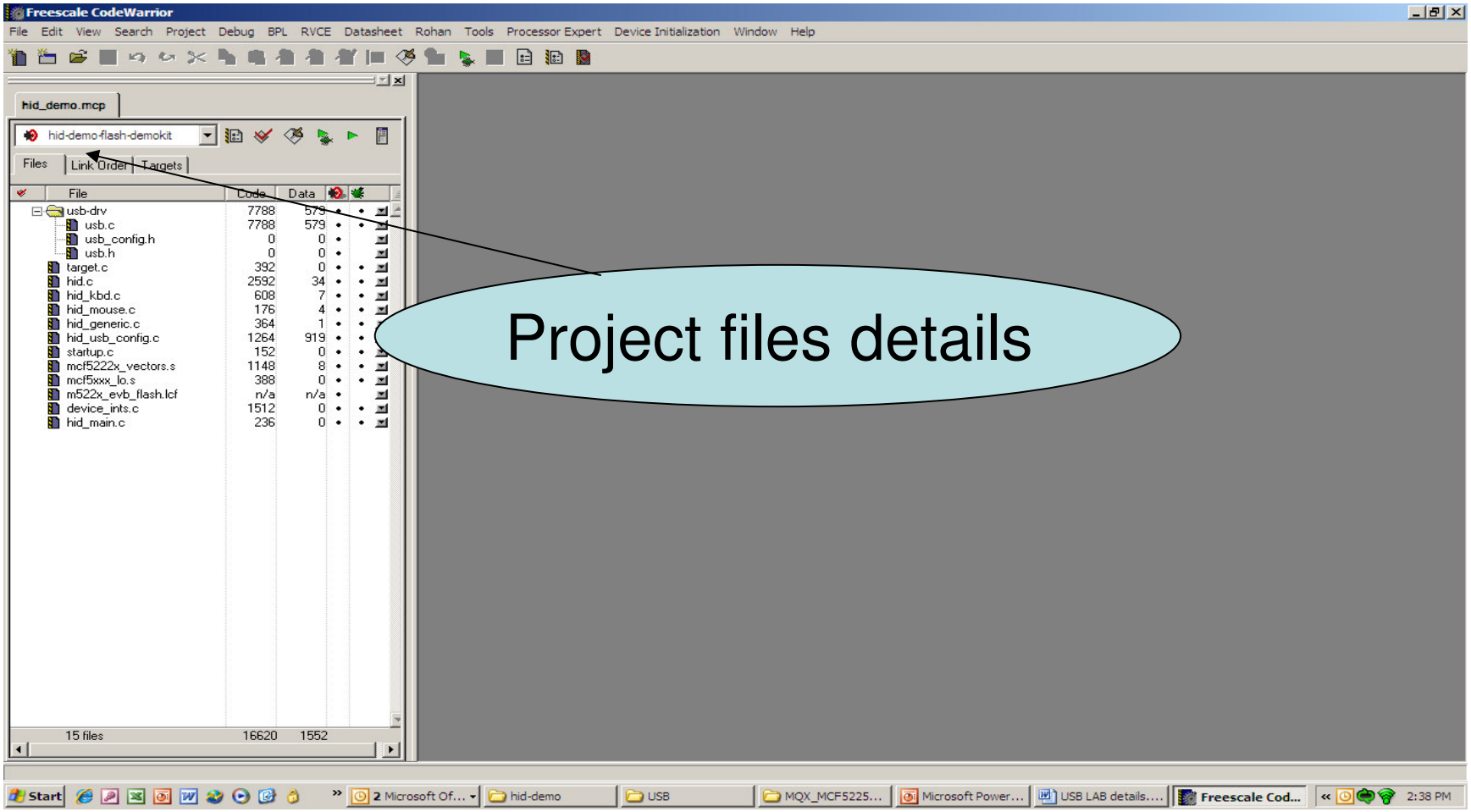
- In this Lab the MCF52259 board (USB) will control the PC curser.
- Install the “Coldfire_USB_Lite.exe” file,
- This Exe will install the required project files for our Labs.
- Open the codewarrior for Coldfire. - GO through the code warrior features
- Open the first project which is located in the installed folder.

- **D:\Profiles\r63554\Desktop\USB for MCF52259\usb-peripheral\projects\CodeWarrior\mcf52223\hid-demo**

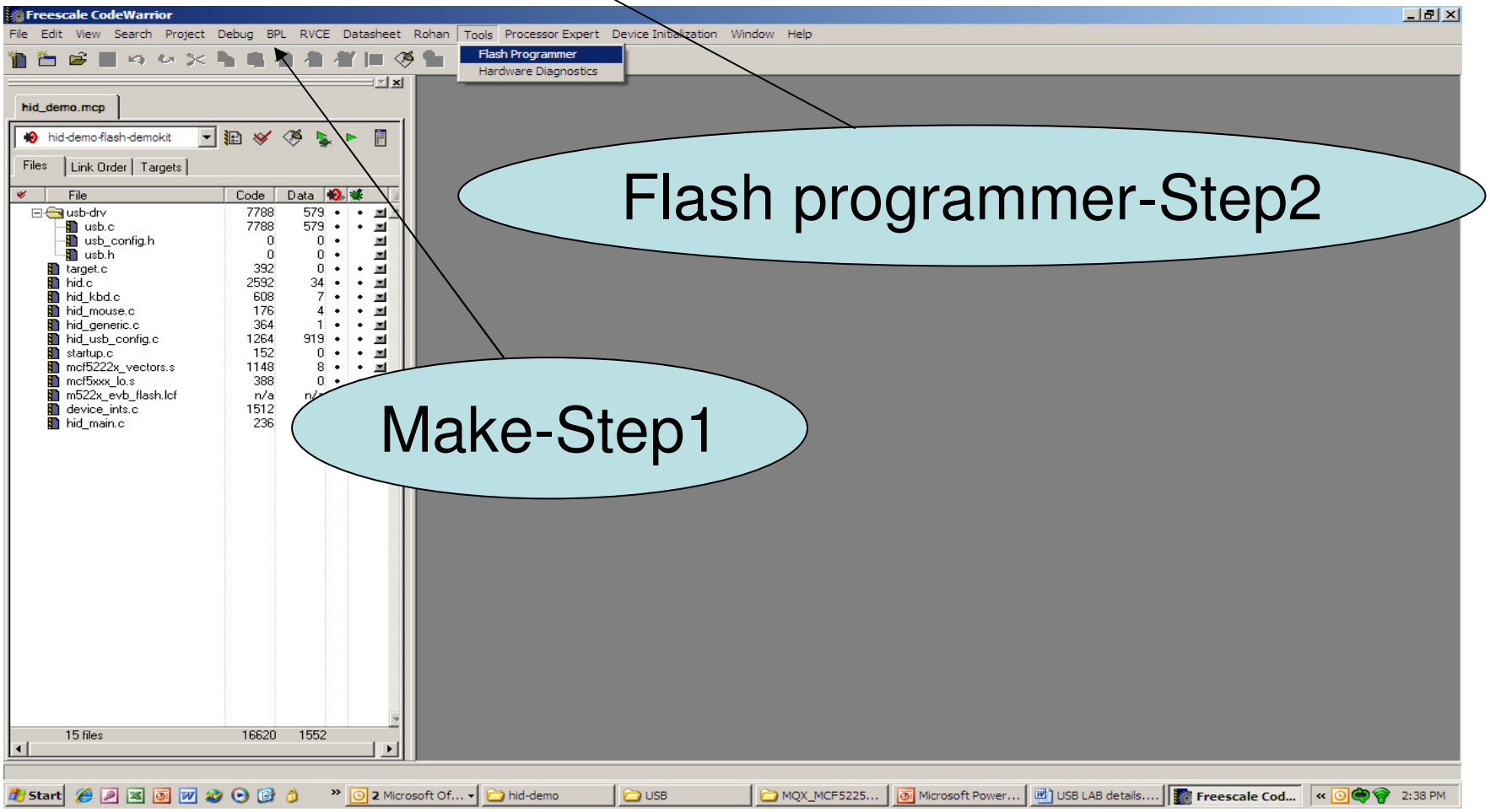
Project file - HID



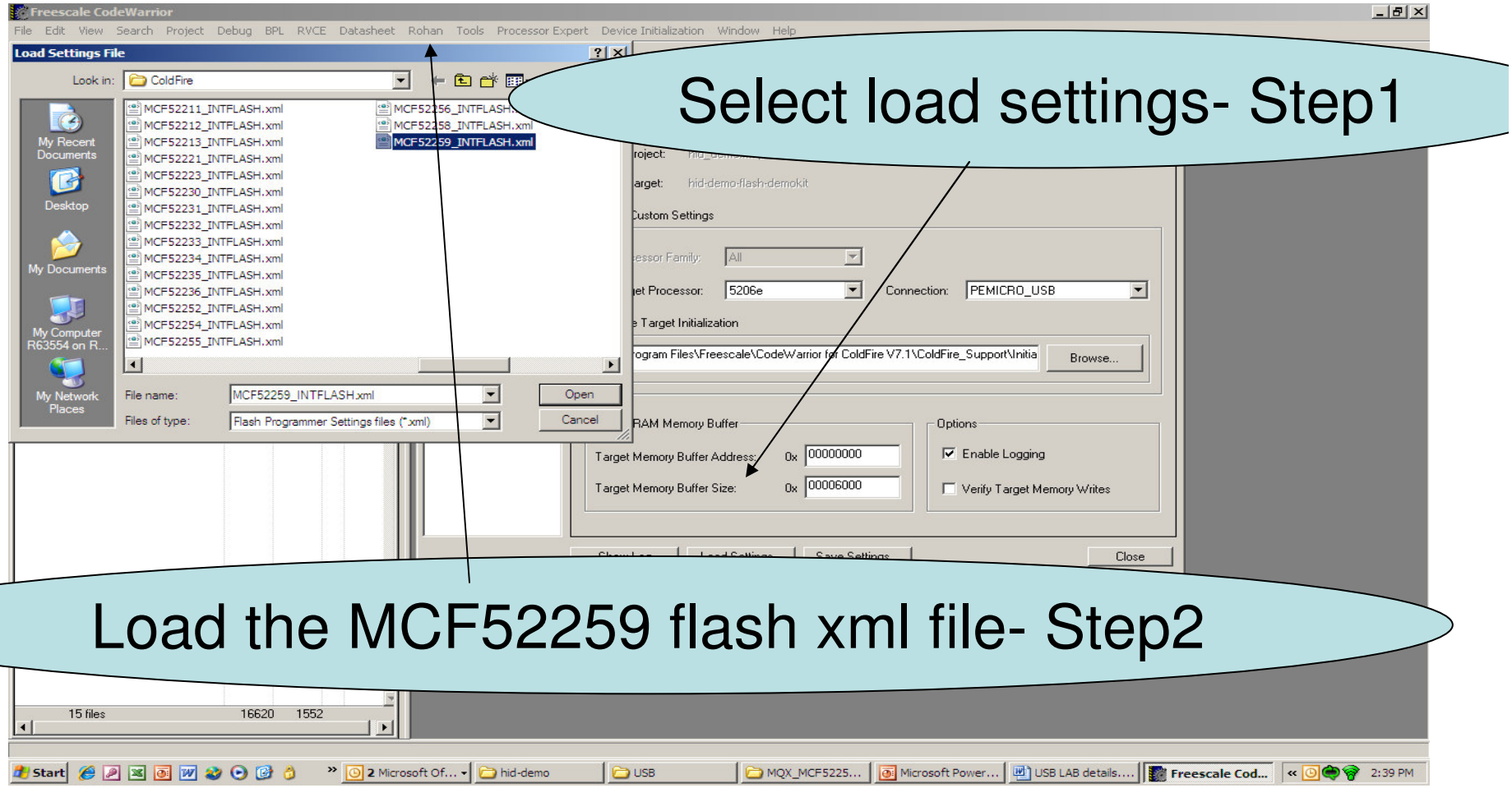
Project file details



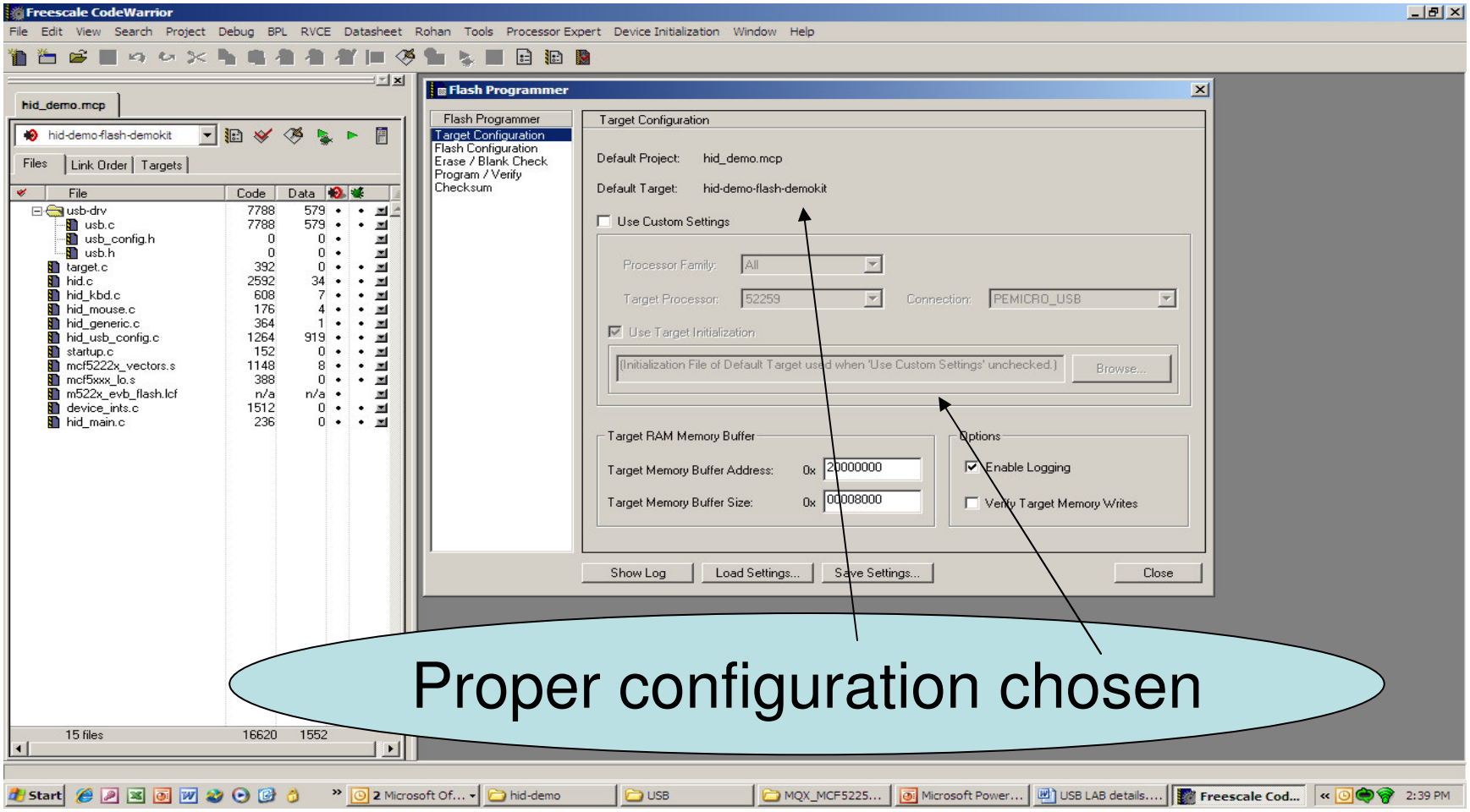
Compiling and flash programming



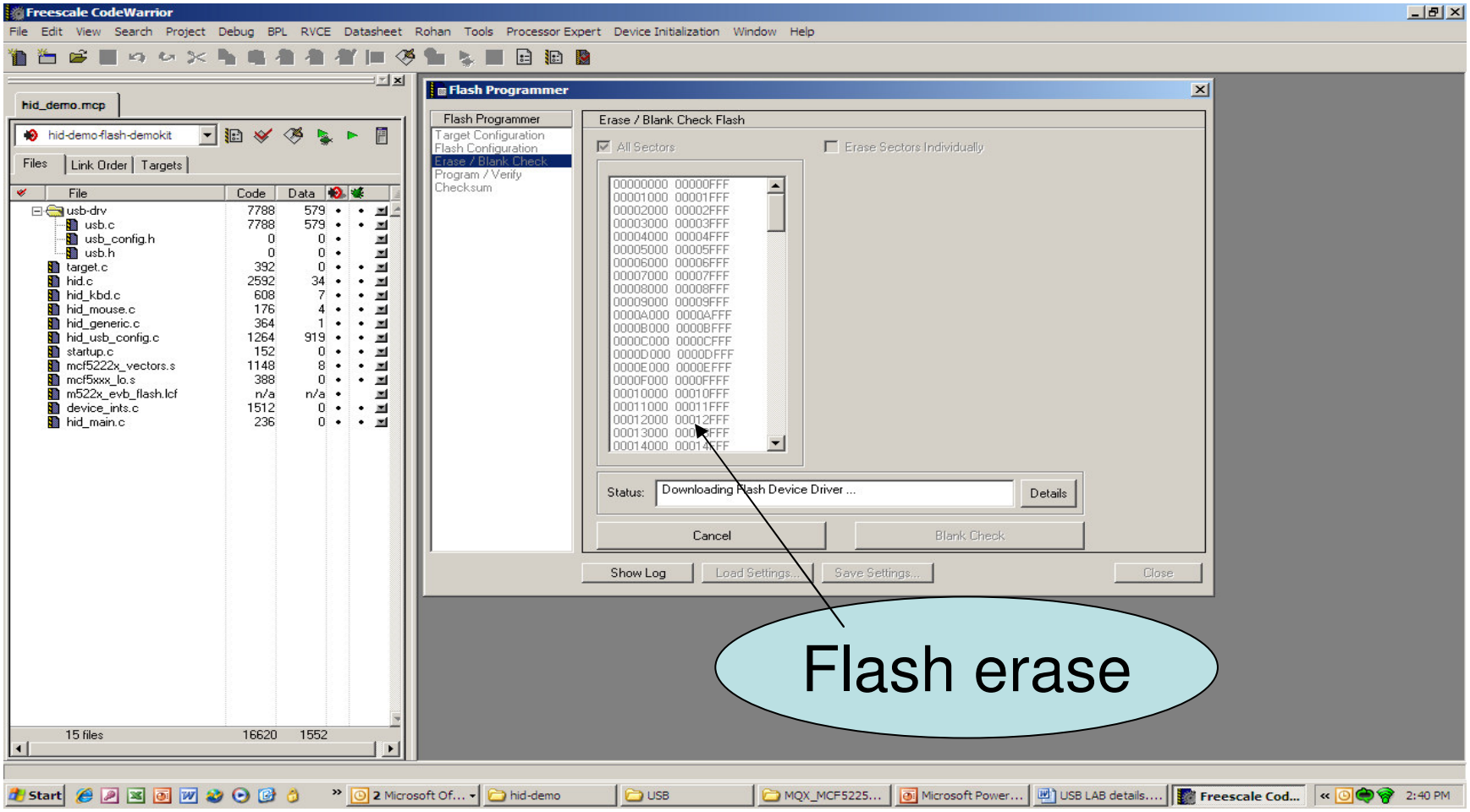
Flash selection



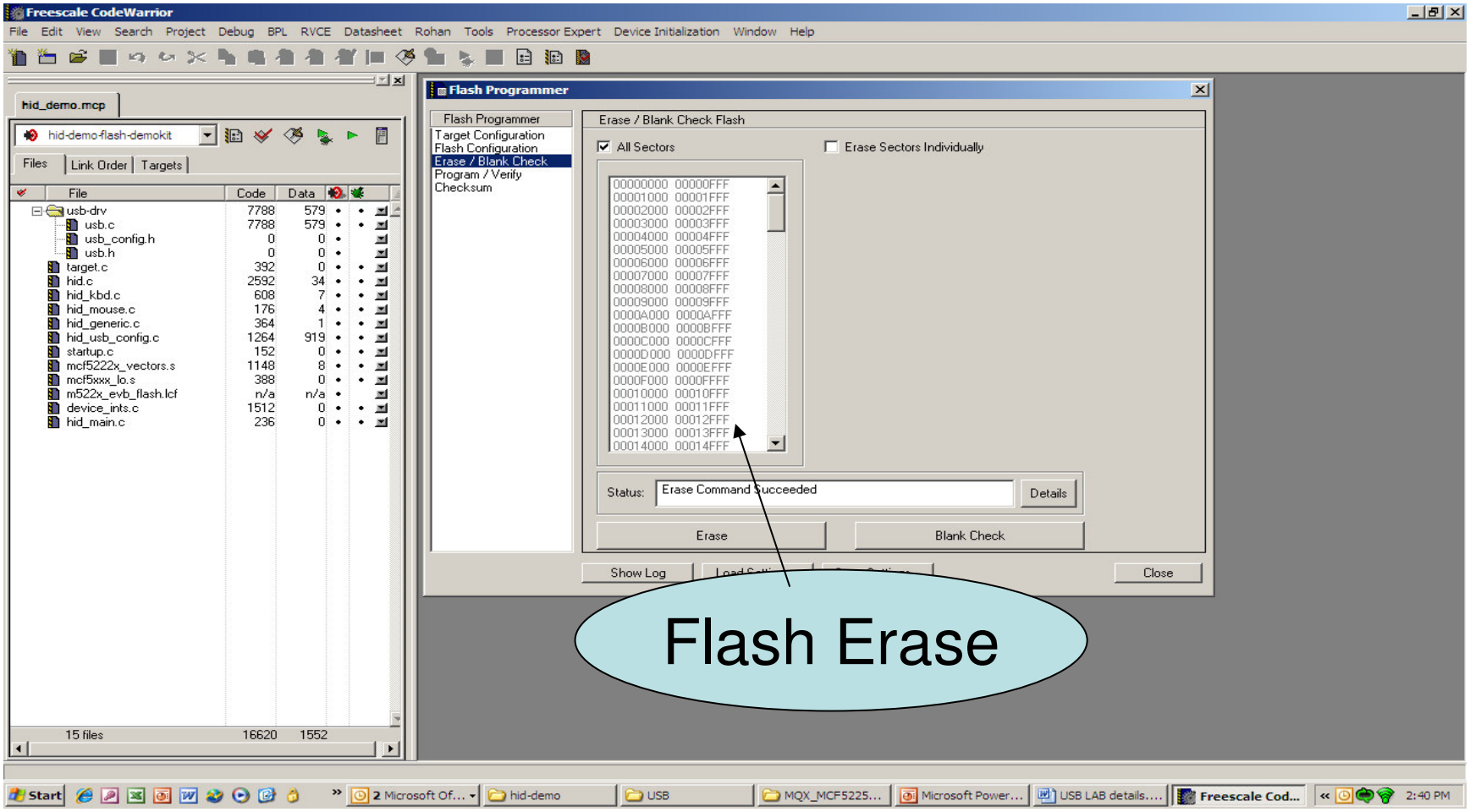
Flash selection



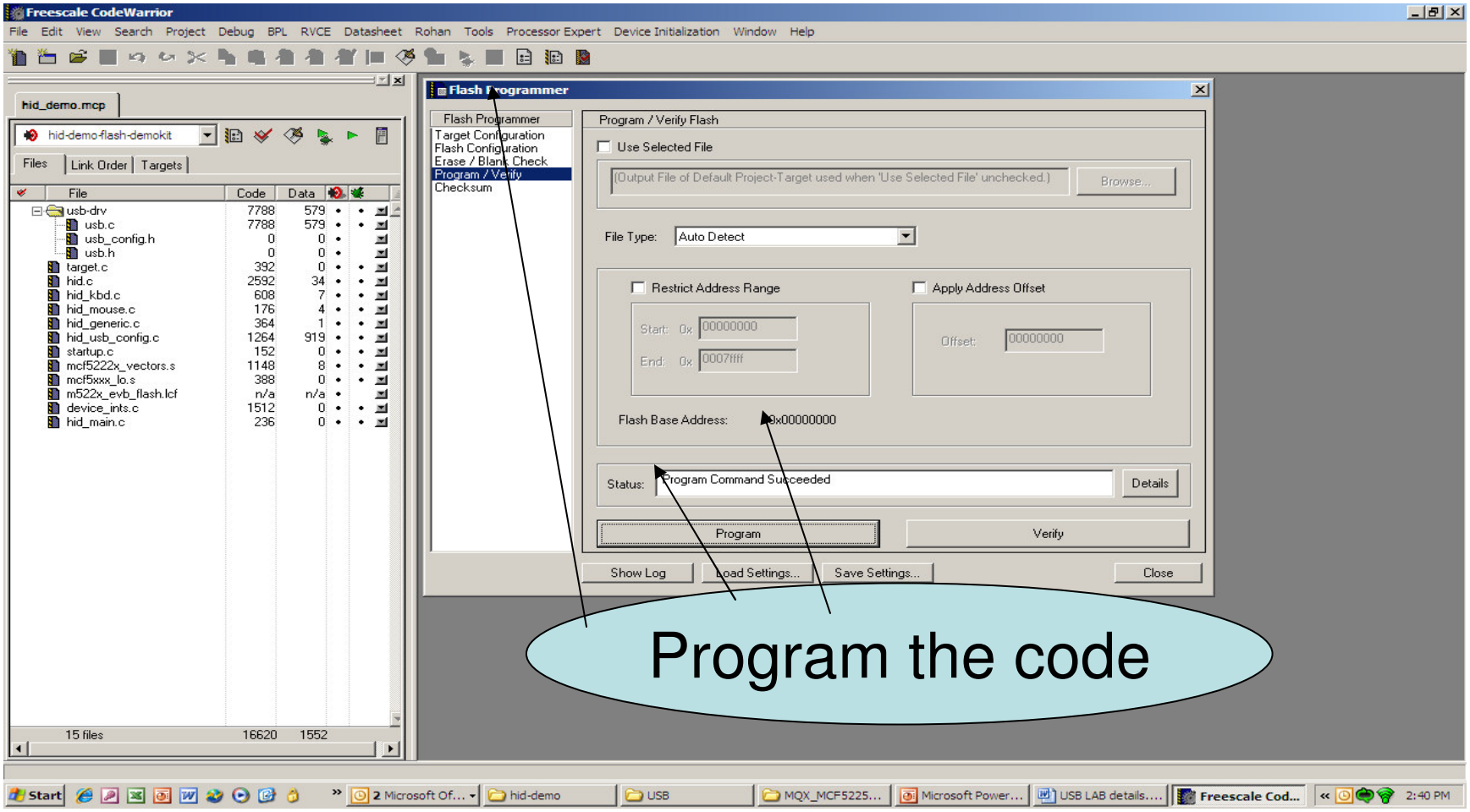
Flash Erasing



Flash Erasing



Flash programming.



Output of the Lab

- Insert the USB Cable to MiniA connector on the peripheral board and other side to your system USB port.
- Reset the board.
- Look at the curser on your system, it should be oscillating.

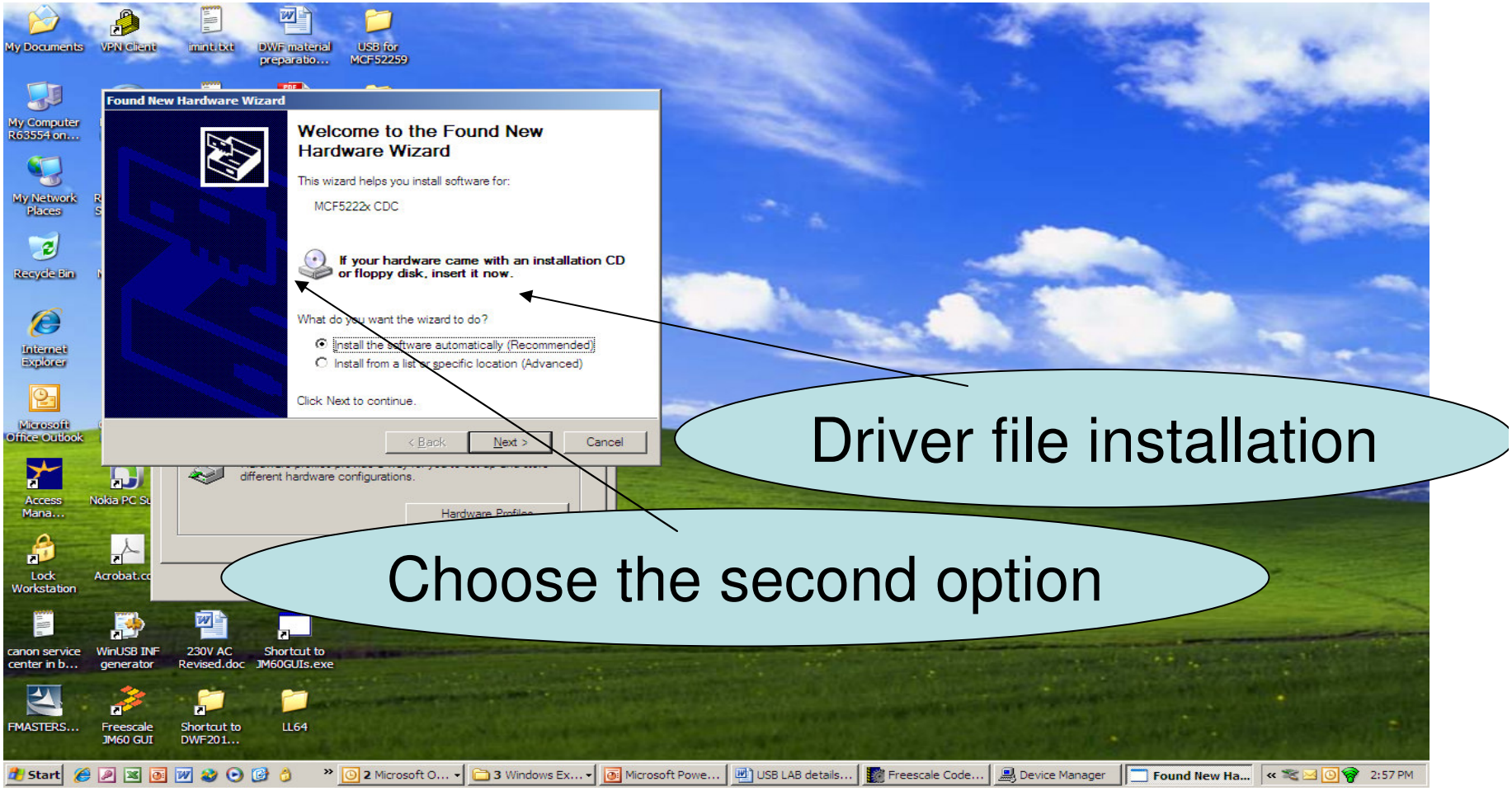
LAB2

semiconductor

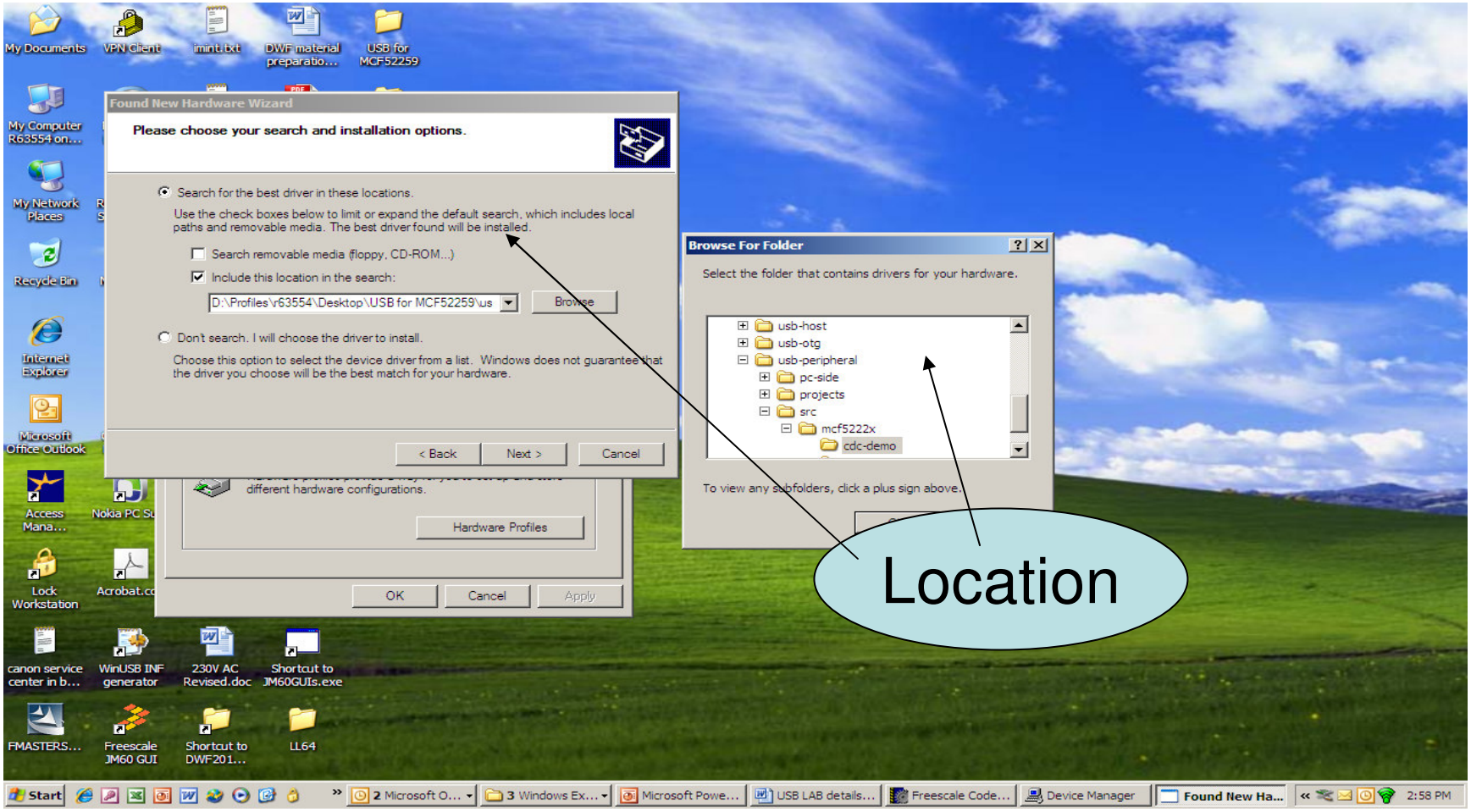
LAB2-CDC

- This LAB will demonstrate the USB to serial and visa versa communication.
- Open the project file from the installed folder.
  - **D:\Profiles\r63554\Desktop\USB for MCF52259\usb-peripheral\projects\CodeWarrior\mcf52223\cdc-demo**
- Follow the Steps which we have followed in the LAB1 for flash programming.
- Insert the USB cable to MiniA connector and other side of the cable to system USB port.
- Connect the serial cable to board serial port & system serial port, (processor board)

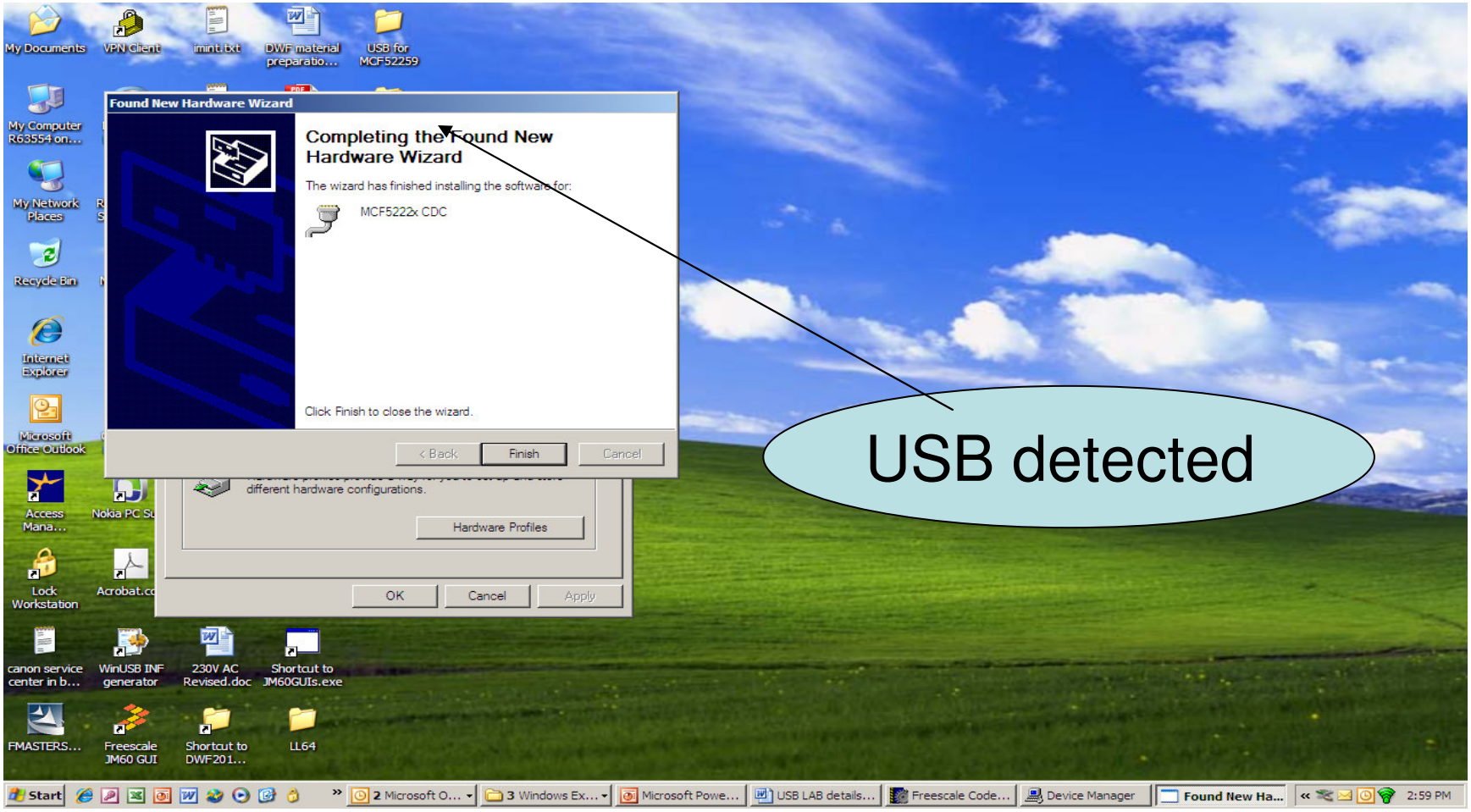
PC side drive file installation



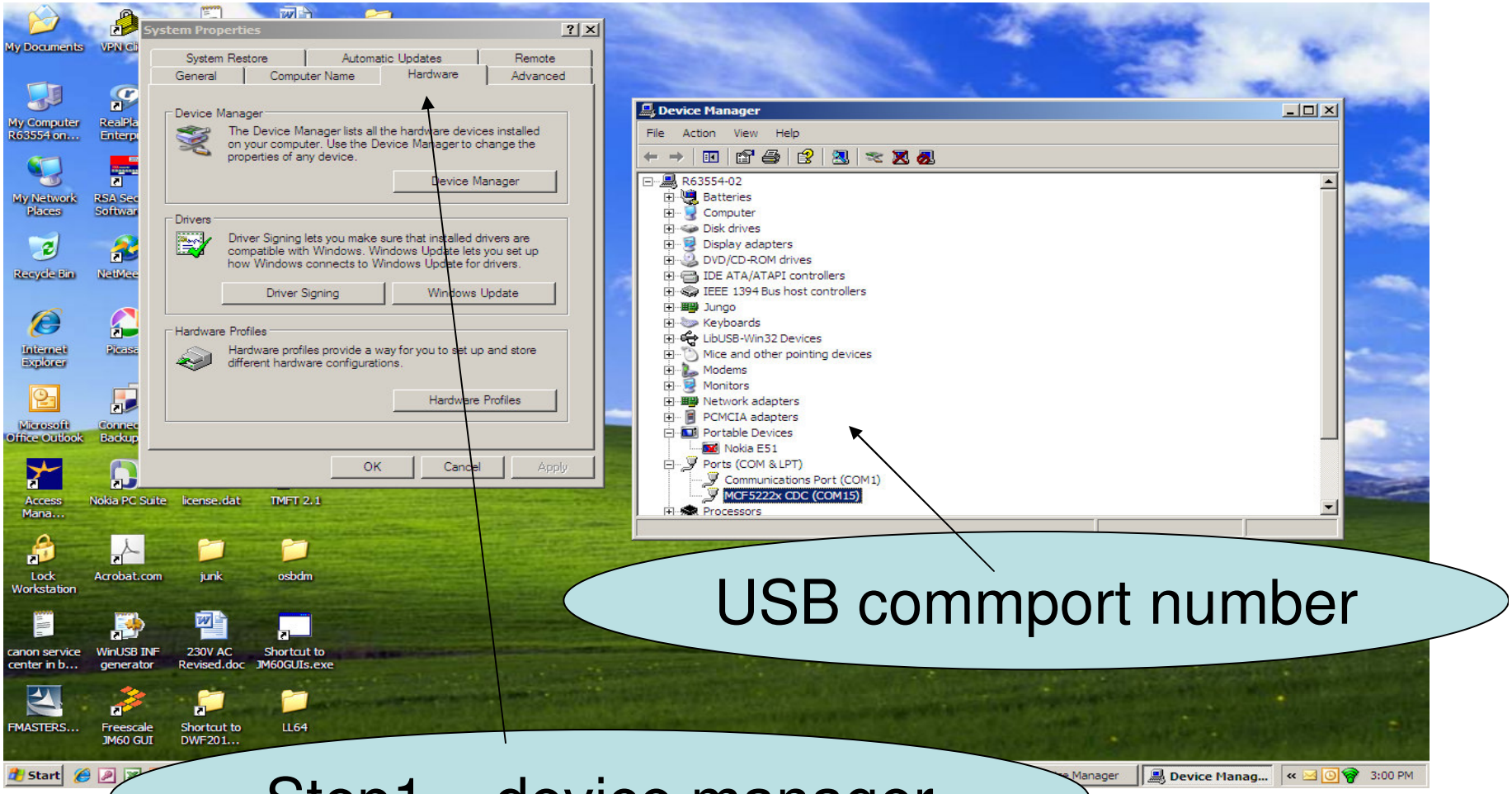
PC-side driver file installation



USB driver


Found New Hardware Wizard
Completing the Found New Hardware Wizard
The wizard has finished installing the software for:
MCF5222x CDC
Click Finish to close the wizard.
< Back
Finish
Cancel
different hardware configurations.
Hardware Profiles
OK
Cancel
Apply
USB detected
My Documents
VPN Client
imint.txt
DWF material preparatio...
USB for MCF52259
My Computer R63554 on...
My Network Places
Recycle Bin
Internet Explorer
Microsoft Office Outlook
Access Mana...
Nokia PC Su
Lock Workstation
Acrobat.cc
canon service center in b...
WinUSB INF generator
230V AC Revised.doc
Shortcut to JM60GUIs.exe
FMASTERS...
Freescale JM60 GUI
Shortcut to DWF201...
LL64
Start
2 Microsoft O...
3 Windows Ex...
Microsoft Powe...
USB LAB details...
Freescale Code...
Device Manager
Found New Ha...
2:59 PM

COM Port at System side



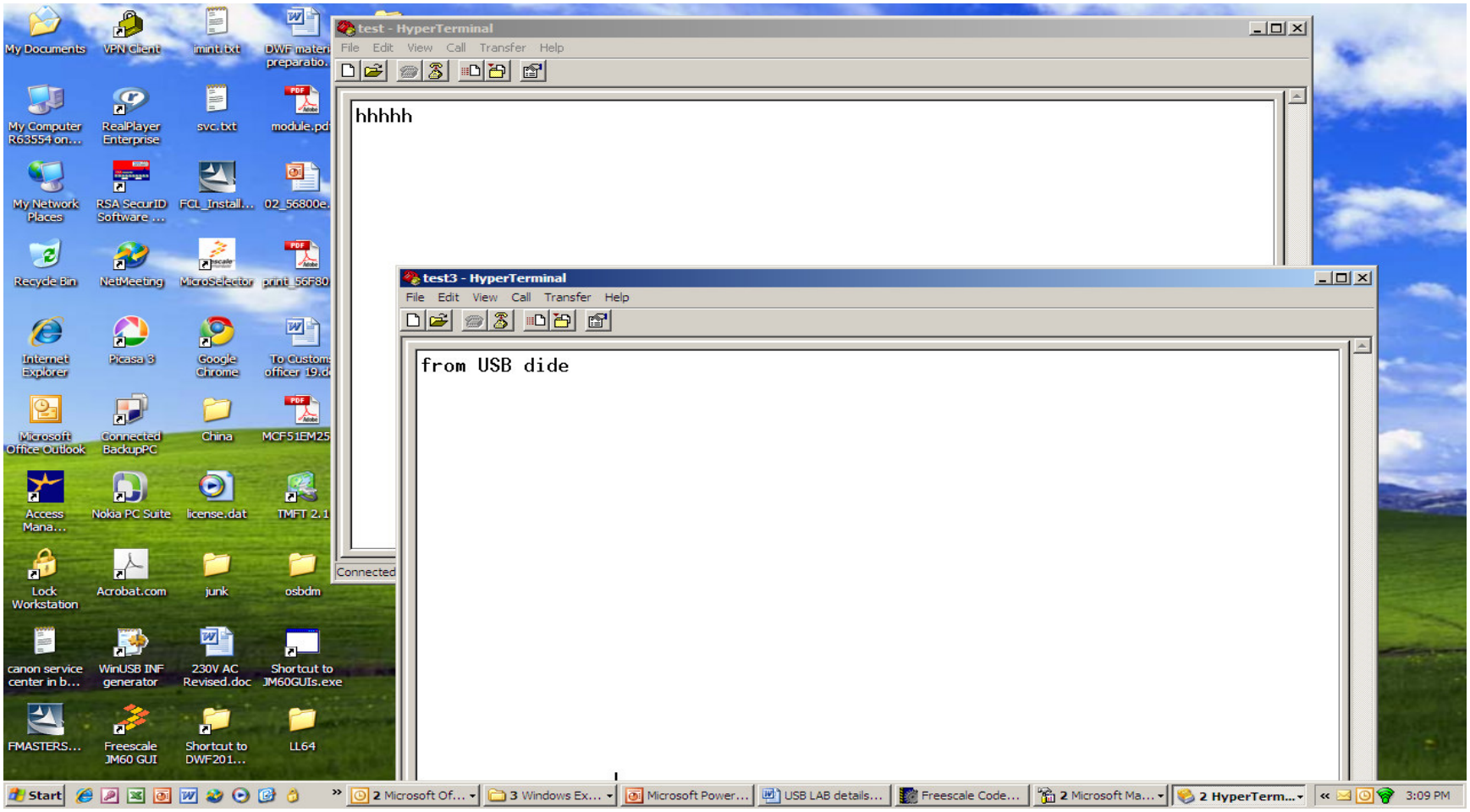
Hyper terminal for USB

- Open the Hyper terminal
- Select the USB port.
- Set the baud rate -19200
- Data bit -8
- Parity -None
- Stop bit- NO

Hyper terminal for UART

- Open one more Hyper Terminal
- Set the Baud rate 19200
- Type the character in one terminal, This shold be displayed in other terminal and visa versa.

Working


test - HyperTerminal
File Edit View Call Transfer Help
hhhhh
Connected
test3 - HyperTerminal
File Edit View Call Transfer Help
from USB dide
My Documents
VPN Client
imint.txt
My Computer R63554 on...
RealPlayer Enterprise
svc.txt
My Network Places
RSA SecurID Software ...
FCL_Install...
Recycle Bin
NetMeeting
MicroSelector
Internet Explorer
Picasa 3
Google Chrome
Microsoft Office Outlook
Connected BackupPC
China
Access Mana...
Nokia PC Suite
license.dat
Lock Workstation
Acrobat.com
junk
osbdm
canon service center in b...
WinUSB INF generator
230V AC Revised.doc
Shortcut to JM60GUIs.exe
FMASTERS...
Freescale JM60 GUI
Shortcut to DWF201...
LL64
Start
2 Microsoft Of...
3 Windows Ex...
Microsoft Power...
USB LAB details...
Freescale Code...
2 Microsoft Ma...
2 HyperTerm...
3:09 PM

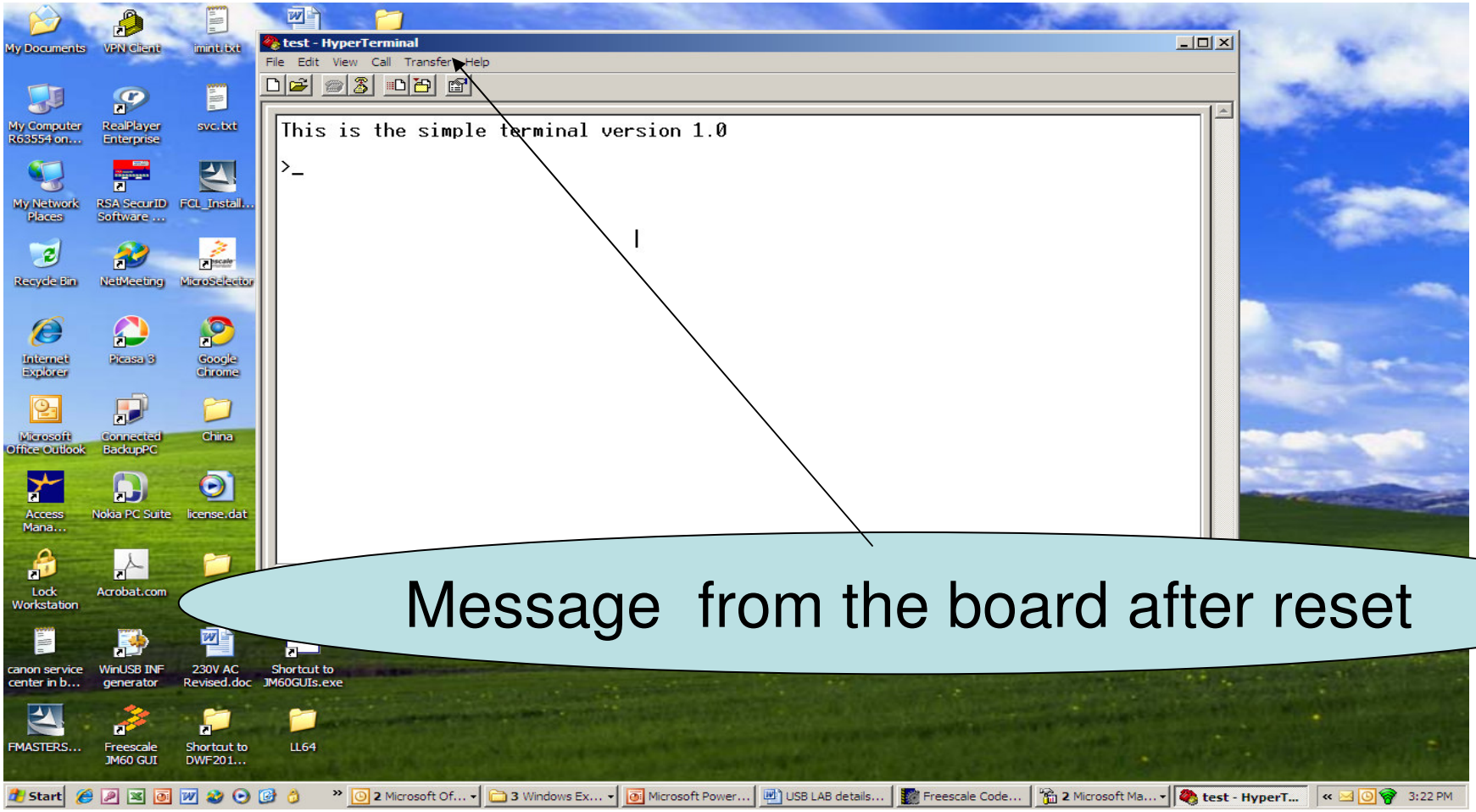
LAB3

semiconductor

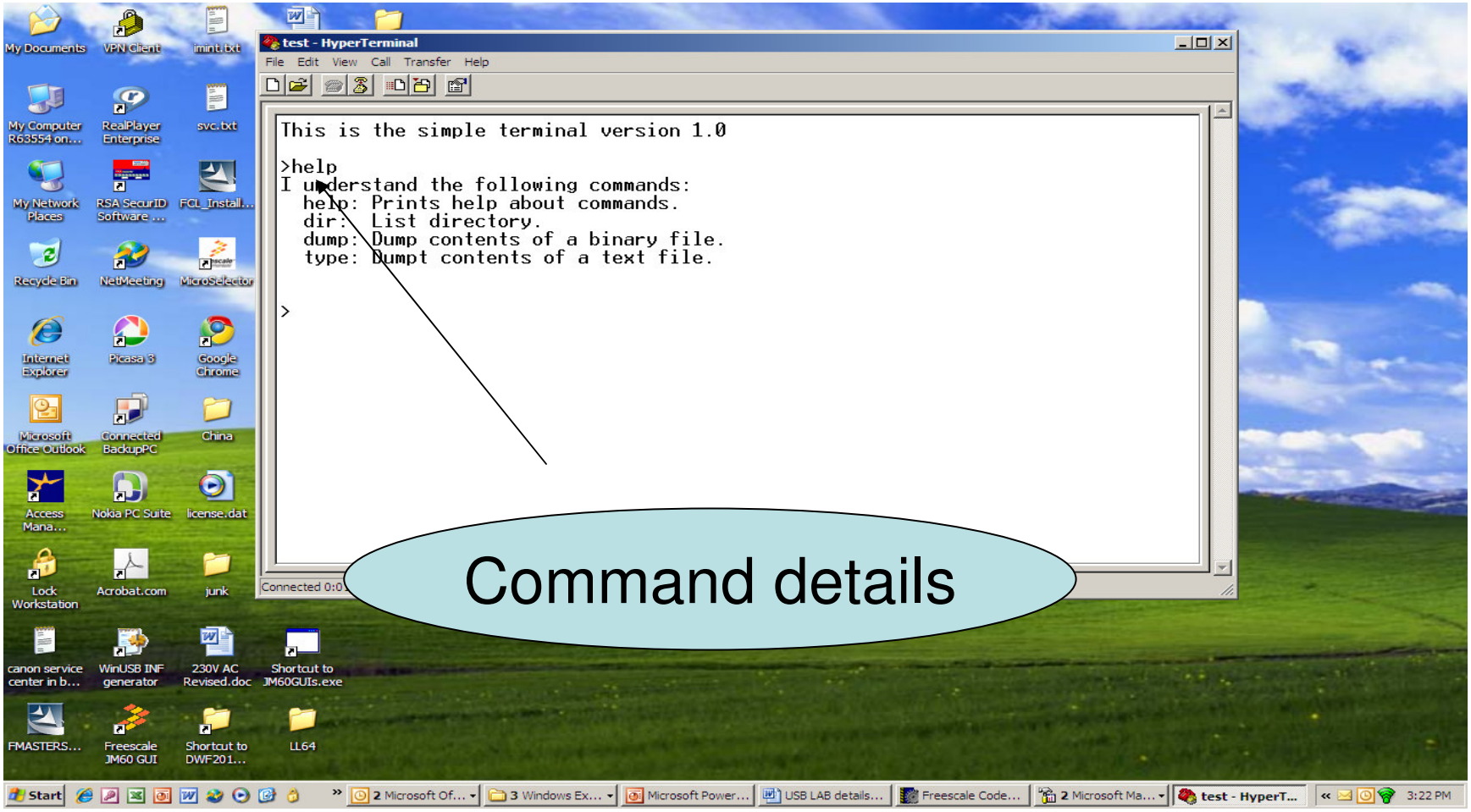
LAB3 – MSD(HOST)

- In this LAB will achieve reading and writing to thumb drive. This will demonstrate the Host functionality.
- Open the Lab from the Installed folder.
- **D:\Profiles\r63554\Desktop\USB for MCF52259\usb-host\projects\CodeWarrior\mcf52223\mass-storage**
- Follow the same steps followed in LAB2 for flash programming.
- Connect the serial cable to system,
- Open the Hyper terminal, select the COM1 and do the Baud rate setting - 115200.
-

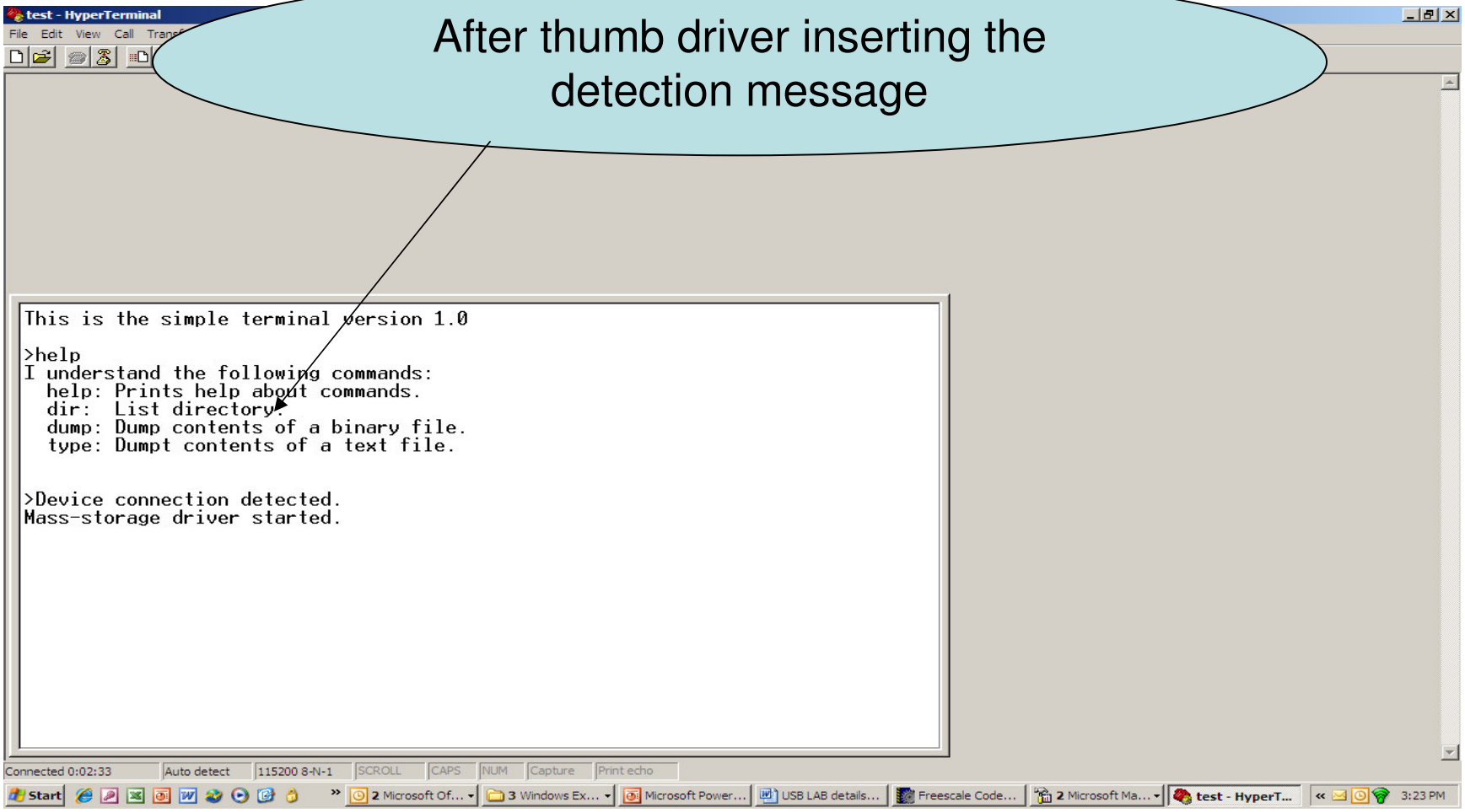
Reset the board



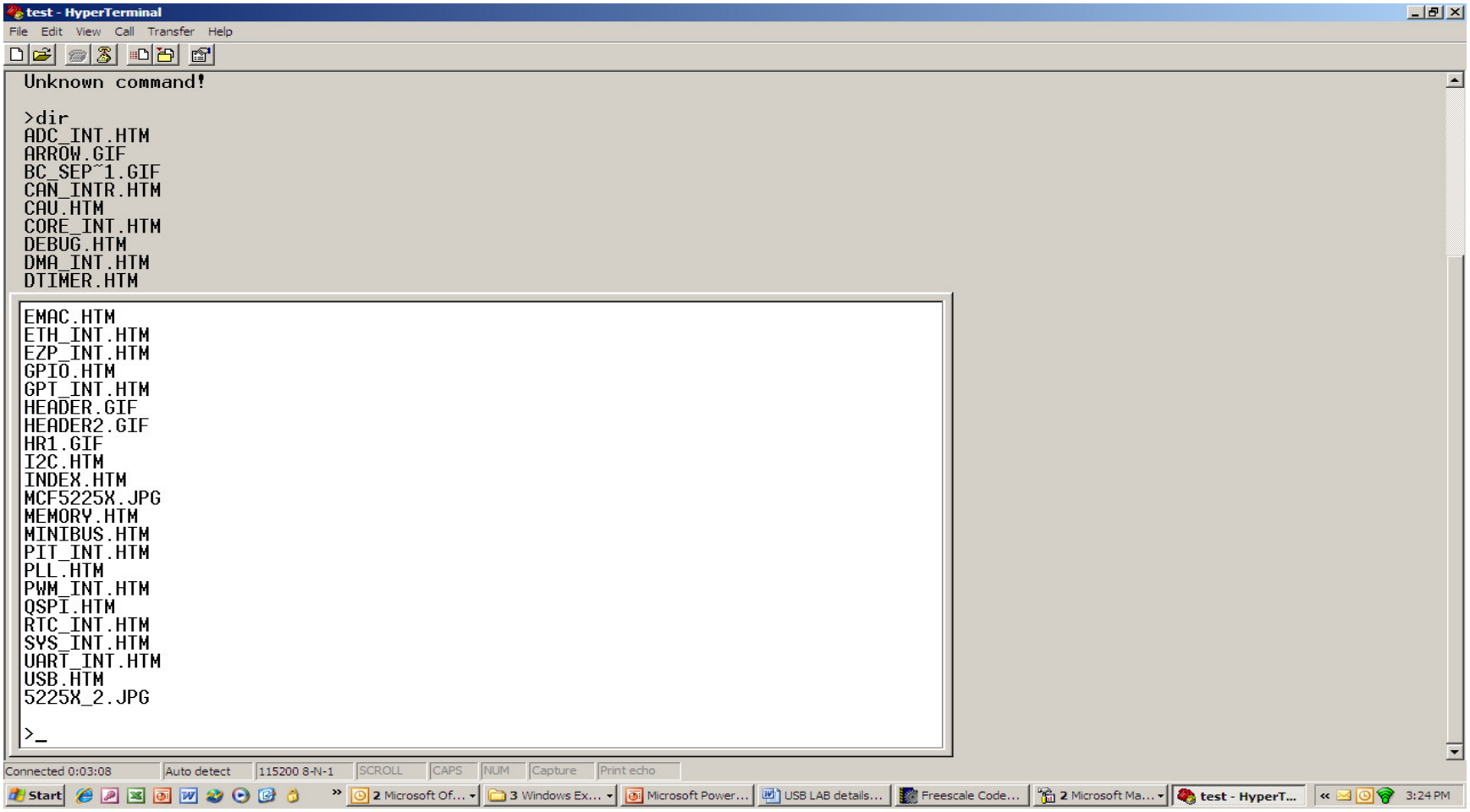
Type Help


test - HyperTerminal
File Edit View Call Transfer Help
This is the simple terminal version 1.0
>help
I understand the following commands:
help: Prints help about commands.
dir: List directory.
dump: Dump contents of a binary file.
type: Dumpt contents of a text file.
>
Command details

USB thumb driver detected



Thumb drive contents.



- AN3560.pdf page 21– for the USB operation


freescale™
semiconductor