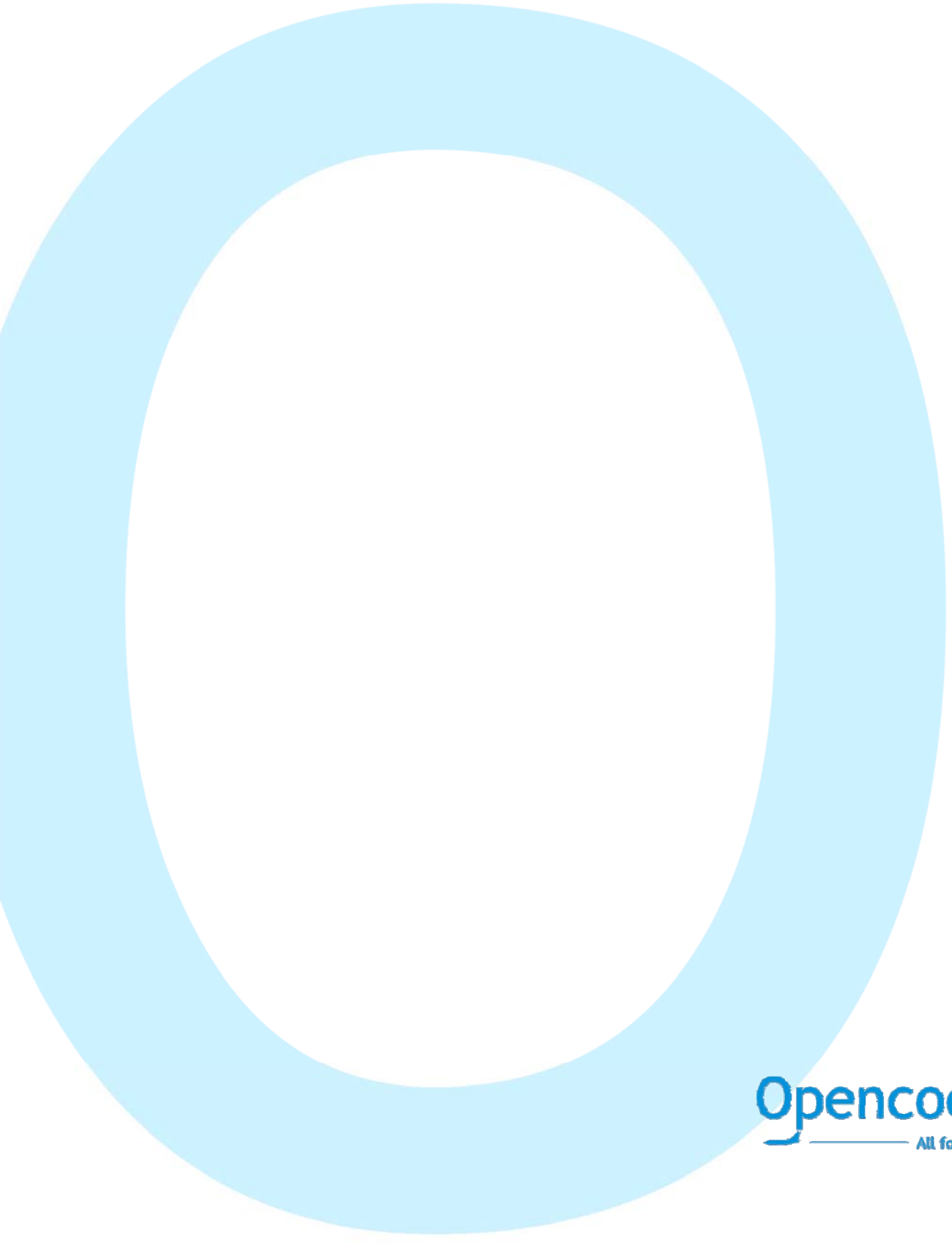
OPENCOCKPITS IOCard USBSTEPPER

INSTALLATION AND USER'S MANUAL



Opencockpits
All for your cockpit

INTRODUCCION

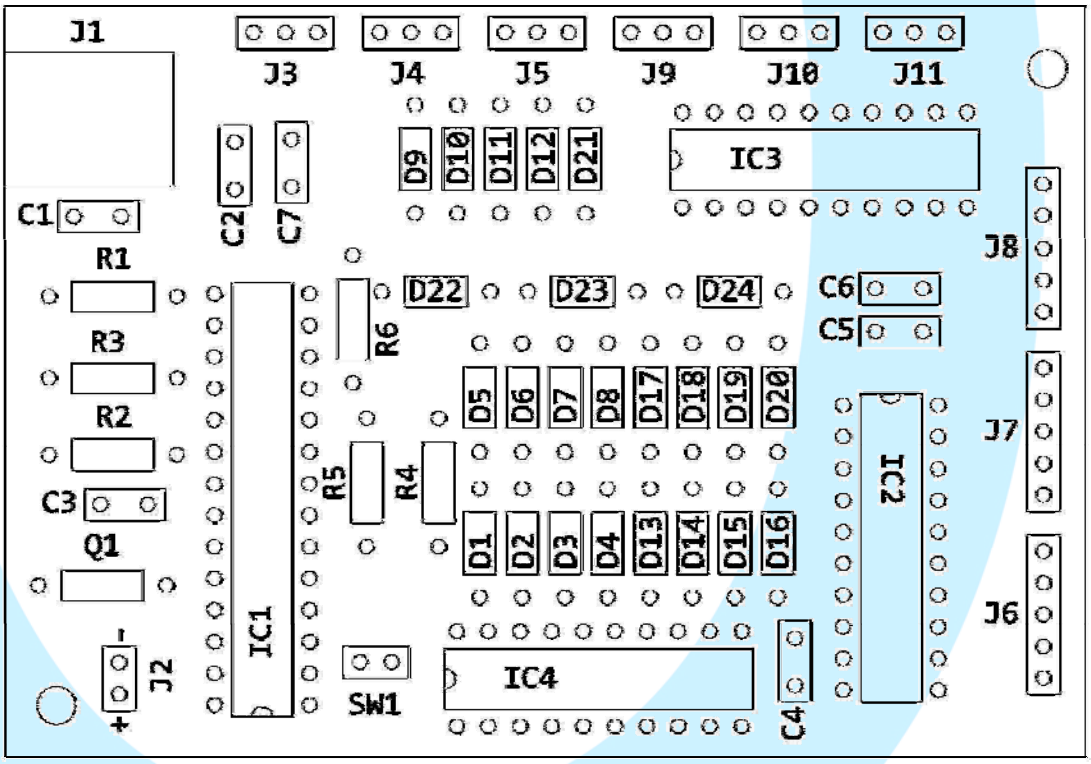
This card allows to manage up to 3 stepper motors, both unipolar as bipolar.

Also, this card can easily control the steppers motors that can be used, as example, on gauges that require a turn more than 1 complete lap, as in the case of an altimeter and where a servo isn't enough.

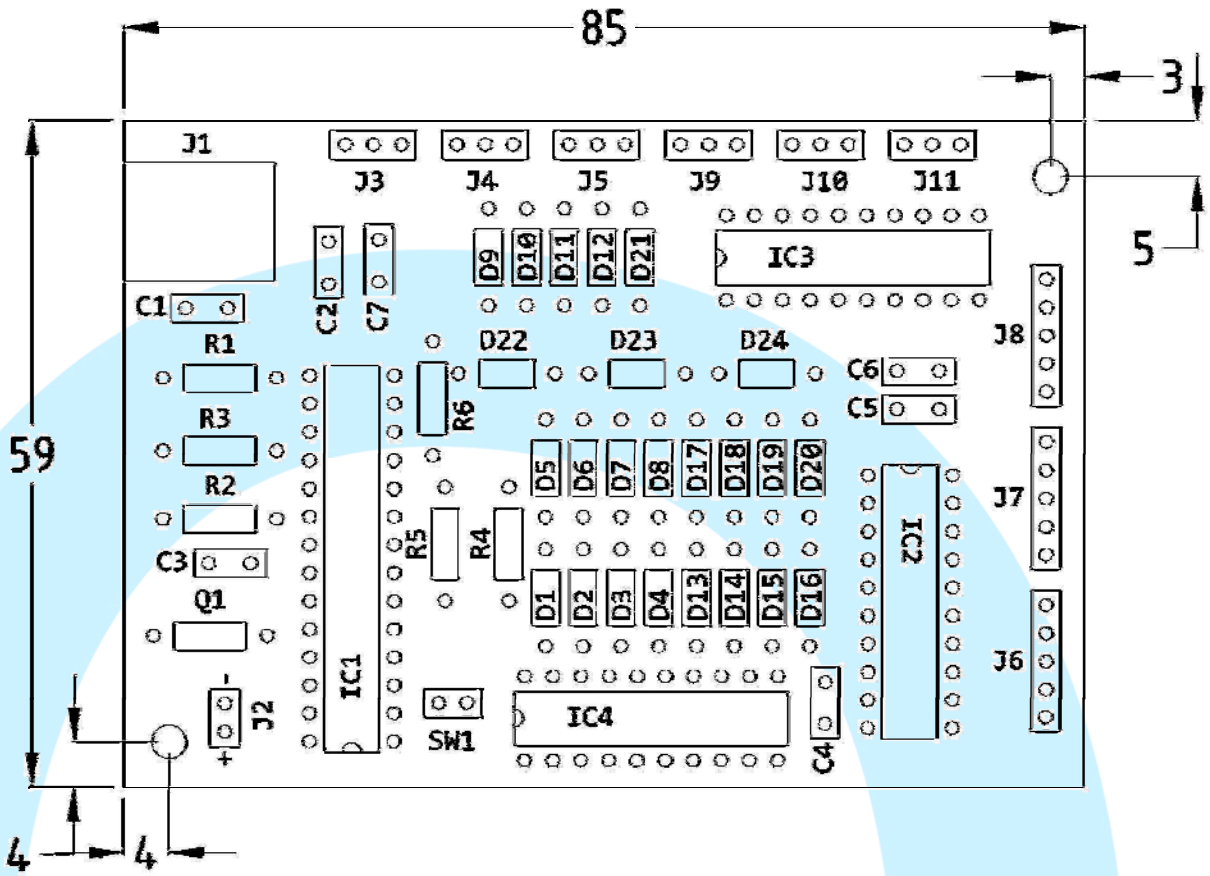
The connection of this card to the PC is through the USB port and when is connected automatically it's detected and installed as an HID device, also for this management uses the IOCP protocol.

COMPONENTS LIST

- C1, C4, C5, C6, C7 = CAPACITORS 0.1Mf
- C2, C3 = CAPACITORS 22Pf
- D1 a D24= DIODES 1N4007
- IC1 = MICROCHIP 16C745
- IC2, IC3, IC4 = IC'S L293E
- J1 = USB CONNECTOR
- J2 = POWER SOURCE CONNECTOR 2 PINS
- J3, J4, J5, J9, J10, J11 = CONNECTORS 3 PINS
- J6, J7, J8 = CONNECTORS 5 PINS
- Q1 = QUARTZ CRYSTAL 6MHZ
- R1 = RESISTOR 100R
- R2, R4, R5, R6 = RESISTOR 10K
- R3 = RESISTOR 1K5
- SW1= RESET 2 PINS



PRINCIPAL MEASURES:



CONNECTORS DESCRIPTION:

- J1 = Allows to connect to the PC directly through the USB port as a HID device.
- J2 = Power source connectorfor to feed the motors (the same power source feed all motors)
- J3 a J5 = Analogic inputs connectors.
- J6 a J8 = Motors connectors (see diagram below)
- J9 a J11 = Position sensors connector (see diagram below)

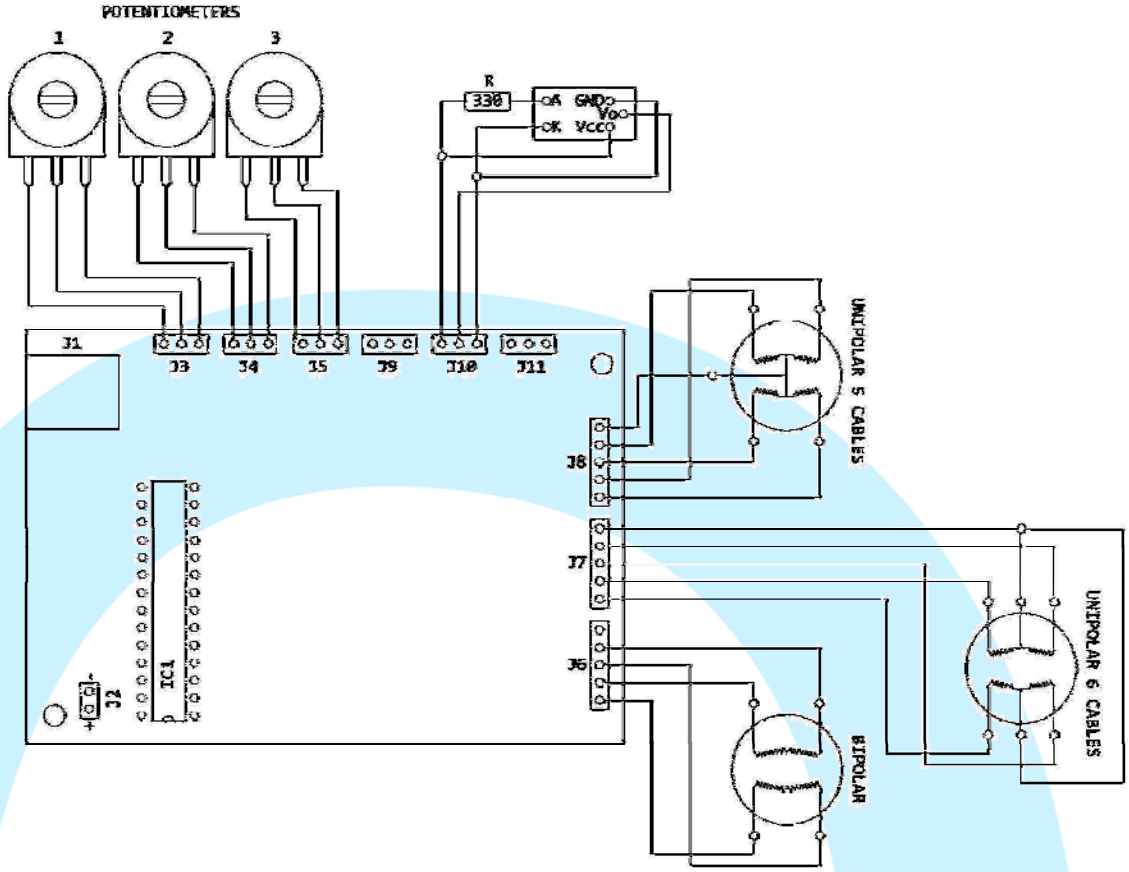
CONNECTIONS DIAGRAM

The card connection is extremely simple, for the potentiometers we have 3 pin connectors (J3 to J6) and are connected as we see in the diagram.

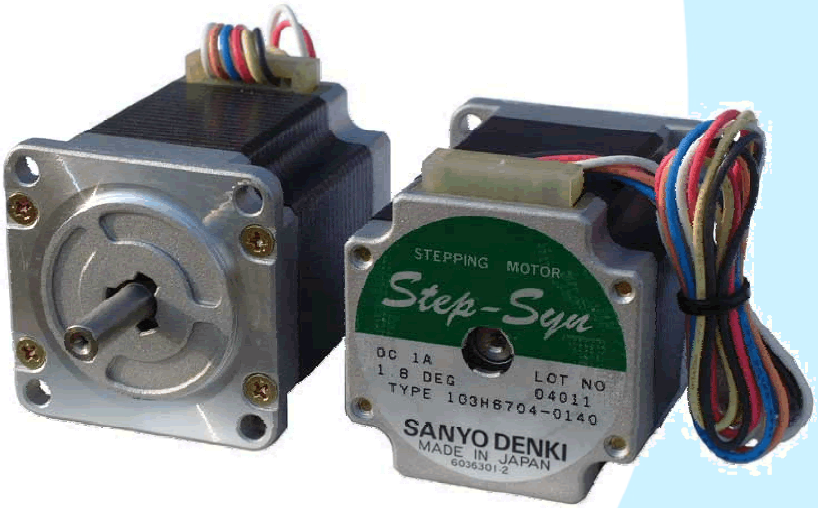
For motors we have from J6 to J8 connectors, connecting the motors depending on the type that we use (see motors section below).

For bipolar motors, use pins 1 and 3 for a coil and pins 2 and 4 for the other coil, leaving the pin 5 free. To connect the unipolar motors we will use the same pins but in this case we will use the pin 5 to connect the common of the coils, in the case of 5 cables only have a common wire, but in the 6 cables type, we shared together and connect the two as before, on the pin 5.

The sensors also are connected to the connectors J9 to J11 in the way that we can see in the diagram below, so that we can control the number of steps in each lap (see sensors section below).



STEPPERS MOTORS

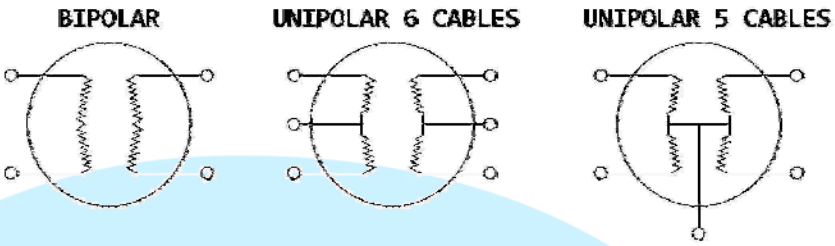


The main feature of the stepper motor is being able to move one step at a time, each pulse that we send to it.

These steps may vary according to the degree that moves in each pulse, that goes from the 90°, to the smaller ones, that will move only 1.8° on each step, so, with the latter that can give us more precision, this means that to turn around (360°), for the first case we would need only 4 steps (90° x 4 = 360°), being necessary to the second 200 steps (1.8° x 200 = 360°).

These motors are basically formed by a rotor on which are applied several permanent magnets and a number of exciting coils. The coils are part of the stator and rotor is a permanent magnet. All the excitement of the coils must be controlled externally using the appropriate driver.

Depending on the configuration of stator windings will have basically two different types of stepper motors:



BIPOLAR:

They usually have four wires that correspond to the ends of the two coils that form (see explanatory drawing above)

UNIPOLAR:

These usually have 5 or 6 wires, depending on the internal wiring of common wire coils (see picture above), note that common for 6-wire motors can also join externally, so that the connection is 5 cables also.

For the USBStepper card is no problem control any of the two types, both bipolar and unipolar, just to be properly configured the connection of the wires.

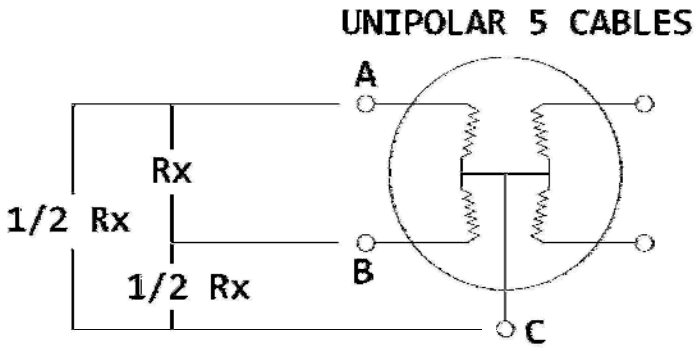
It should be noted that such mechanical devices are, the stepper motor must overcome certain inertia, therefore the duration and frequency of the pulses applied is a very important point. It's this sense the engine must reach the step before we send it the next pulse, for this reason, if the pulse rate is very high, the engine may react as follows:

- Do not make any movement at all
- Vibrates but without turning
- Turns in a erratic way
- Or get to rotate counter to the desired

If we don't have data sheets of the motors, because they are recycled or recovered, or because they are new but we have no data, it is possible to determine the distribution of wire connections to the coils and the common wire, in the case of 5 or 6 wires unipolar motors, following a small step instructions below:

For 6-wire unipolar motors, it is better to merge the two common wire (usually the same color) prior to performing the tests.

Use a tester to check the resistance between pairs of wires, the common wire will be the only one who has half the resistance. This is because between the common and any other cable has only one coil, in contrast between the pairs of cables there are always two coils (see diagram of testing below)



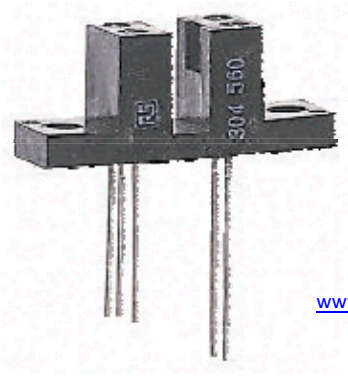
In this case, if we measure with the tester the resistance between points A and B, we get x resistance, but instead we measure between points A and C or B and C, we get only half the resistance, which would indicate that the wire C is common.

For bipolar motors (usually 4 wires), the identification is easier, simply by measuring the resistance between pairs, which form part of the same coil will have continuity (low resistance), being the other two wires ends the other coil. To determine the polarity of these coils, we'll do it just trial and error method, reversing the wiring position if the rotation is not as expected.

Remember:

- A motor with 5 cables is almost certainly UNIPOLAR.
- A motor with 6 cables is also almost certain, UNIPOLAR, but with 2 common wires (may be the same color)
- A motor with only 4 wires is commonly BIPOLAR

SENSORES DE POSICION



www.amidata.es refª 304-560

These sensors will serve to USBStepper card know always the position of the shaft when the motor is turning, in this way the shaft, is always in the same position.

The card, to connect to the software, the first thing that does is to start turning the motor to pass up to 2 times the optical sensor. Why 2 times?, Because the first it does is to put an internal counter to 0 and from this time begins to count the number of steps you take to complete the second turn and pass a second time thorugh the optical sensor and knows exactly the number of steps taken by the motor and also the time it takes to complete that lap.

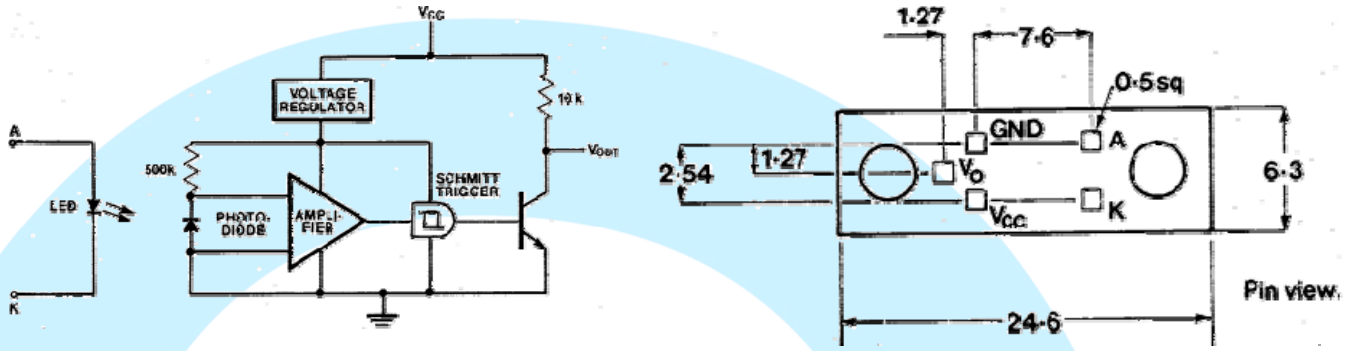
At this point, we know the number of steps of a turn, also known starting position marked by the sensor, then you can calculate any relative position from that starting position, simply keeping track of the steps you take to one side and on the other.

The optical sensors used must be capable to supply voltages TTL (0 and +5 V), so it is recommended that embody the microelectronics to deliver TTL voltages.

These sensors consist of a light emitter and a receiver, so that the voltage becomes 0 when the ray stopped and +5 V when the sensor is not blocked.

For connection, see diagram above.

We recommend for example that can be located with reference 304-560 in www.amidata.es , and this is the scheme and measures of that sensor:



USING SIOC

Make sure you have installed version 3.46 or higher, if not you can download the latest version here:

http://www.opencockpits.com/catalog/info/information.php?info_id=31&cPath=2

Once you have the correct version, the first thing to do is set the parameters on the sioc.ini file to make sure that the card is correctly identified with the Device number as corresponds.

We will edit the entry in the sioc.ini file, so we assign an index of device to each card that we will install, create an entry in the file per connected card, and would be in following format:

USBStepper=XX,YY

Where XX indicates the index number, within our system and YY is the number of device from the USB port where is connected.

For example, if we connect two USBStepper cards with the device numbers 35 and 42 (these numbers can be easily find in the SIOC.exe program itself, since it will provide us information on the card) then would declare the sioc.ini as follows:

USBStepper=1,35
USBStepper=2,42

Isn't a problem to have more IOCards connected in this computer, while they are correctly defined, nor to have Opencockpits modules also connected.

STEPPER MOTORS:

To refer to the number of motor accurately, we must consider the index number we have assigned to each card USBStepper.

Now in SIOC, we must define the output in standard form:

Var *VVVV*, **name** *NNNNNNNNNNN*, **Link USB_STEPPER, device** *DD*, **Output** *S*, **posL** *LLL*, **posC** *CCC*, **posR** *RRR*,**Type** *T*

- VVVV= variable number
- NNNNNNNNNNNNN = variable name *(optional)*
- DD = index number that is defined in ini file *(optional, if we have defines as a number 0, isn't necessary top ut this number in Device parameter)*
- S = motor number 1-3
- LLL = motor speed value 0-255 *(as lower is this value, more speed, but we must be warn not to overspeed the máximum value of motor)*
- CCC = motor step number 0-65535 *(if we left this number in 0,the calibration will be automatic)*
- RRR = maximum number of steps per tenth of a second (0-255) *(this value is tricky because if you put a very high number, overflow the internal buffer where we accumulate orders of positioning and therefore lose the position, if it'is low, the hand down the speed of positioning, so, a number between 3 and 5 is perfect)*
- T = if we define this parameter as an"H", then the steps are in ½ steps, so, then we have the double of steps.

Definition example:

Var 0001, name step_alt, Link USB_STEPPER, Device 1, Output 1, PosL 6, PosC 0, PosR 4, Type H

ANALOGIC INPUTS:

To read the analogic inputs we should use the following format:

Var *VVVV*, **name** *NNNNNNNNNNN*, **Link USB_ANALOGIC, Device** *DD*, **Input#** *EE*, **posL** *LLL*, **posC** *CCC*, **posR** *RRR*

- EE = analogic input number 1-5
- LLL = maximum position to the left.
- CCC = center position of device
- RRR = maximum position to the right.

The rest of parameters will use as on relays definition.

Analogic definition example:

Var 1506, name pot_flaps, Link USB_ANALOGIC, Device 1, Input# 2, posL 1, posC 128, posR 255

SIOC EXAMPLE

We make a gauge for altitude (altimeter) and will mount a stepper motor in the needle of hundreds, with the gear reduction that we deem appropriate, to obtain enough accuracy. Connect the motor and sensor to the card and feed it with the voltage necessary for the motor.

Run SIOC and let the motor to make an auto-calibration, if we run the FS now the needle after calibration should be in the proper position to the height of the plane and follow the actions of the aircraft, so we can make some ups and downs to test it.

```
Var 0001, name alt_fs_h, Link FSUIPC_INOUT, Offset $3324, Length 4
{
  L0 = MOD &alt_fs_h ,1000
  &step_alt = L0 * 0.36
}
Var 0002, name step_alt, Link USB_STEPPER, Device 1, Output 1, PosL 6, PosC 0, PosR 4,
Type H
```

The only change we should do, is the correction of the needle itself, whether to assemble the instrument, the alignment was not perfect, for it would add the degrees of difference to the result to send to the engine:

```
  L0 = MOD &alt_fs_h ,1000
  L1 = L0 * 0.36
  &step_alt = L1 + XXX // XXX = difference degrees
```