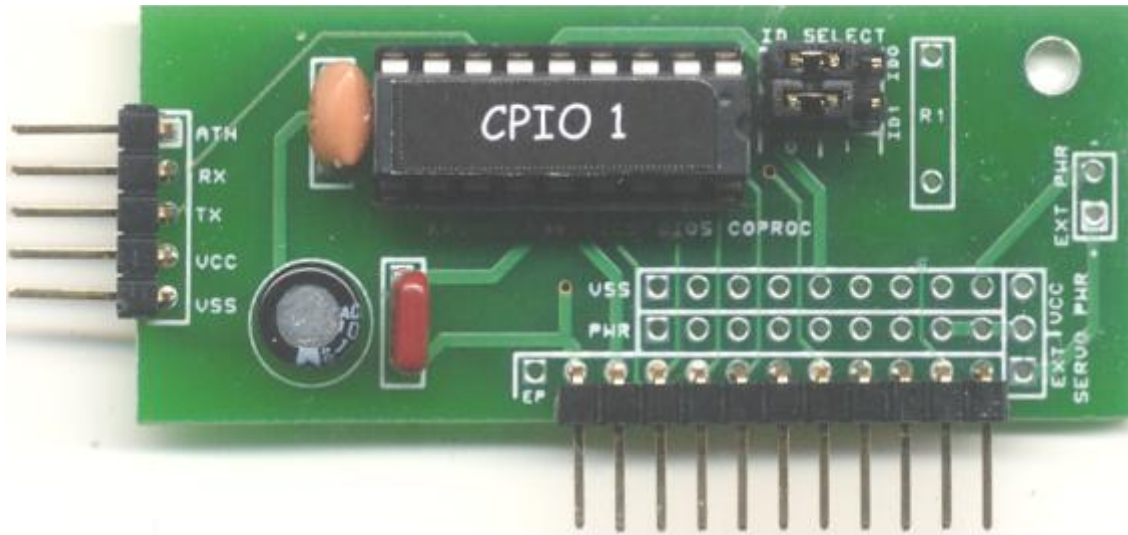


CPIO1

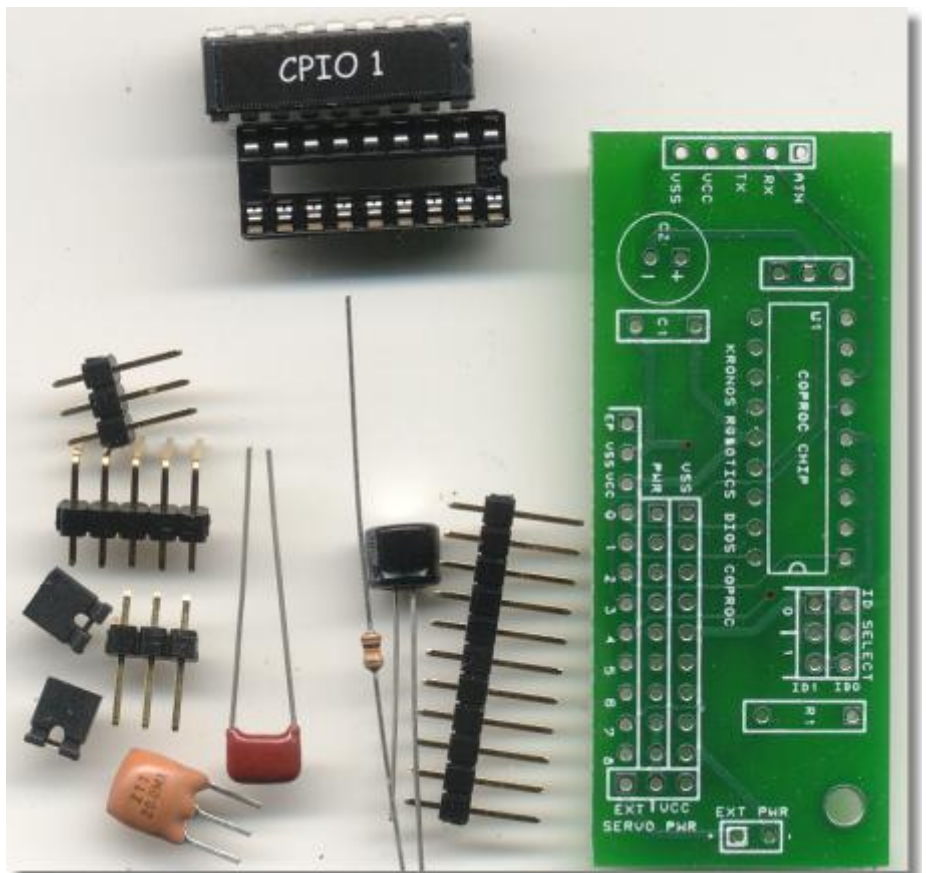
Assembly and Hookup

cpio1 Manual Version 1.0



Your Kit will include the following parts

- Printed Circuit Board
- Assembly and Hookup Manual
- 18 pin Socket
- 5 pin Right Angle Header
- 100uf Capacitor
- .1uf Capacitor
- 20Mhz Resonator
- CPIO1 Chip
- 2, 3 pin headers
- 11 Right Angle pin header
- 10k Resistor
- 2 jumpers



Kronos Robotics
and Electronics



Specifications

CPIO specs

IO Ports 9

Input/Output/low speed pwm (~600hz per channel)

Hardware pwm (optional on port 4)

Max resolution 10 bit

Max frequency 2.5Mhz

Min frequency 1.24Khz

IO speed based on 115200 baud (Faster bus speeds will give higher throughput)

350us (using iowrite access to all 8 channels can be achieved with 1 command)

Jitter <8us (due to Uart requests)

Controller power requirements (does not include servos)

3.0v—5.5v

3ma

Controller Speed

20Mhz

Protocol specs

Async 9600-1250000 baud (programmable)

115200 default

90 byte IRQ driven buffer

Multiple units can be placed on single bus as slaves.

Protocol 2-9 byte variable byte protocol. With floating transmit (slave)

Slave transmit hold time (programmable)

Slave transmit delay (programmable)

Slave Resync hold time (programmable)

Internal error recovery with status indicator

© Copyright 2003

Kronos Robotics

www.kronosrobotics.com

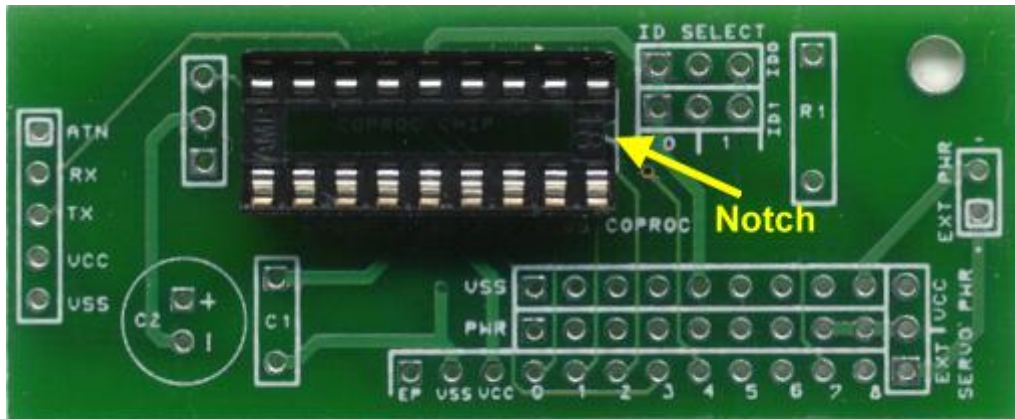
msimpson@kronosrobotics.com

Table of Contents

Specifications	2
Assembly	4
Hookup	6
Protocol Definition	8
Command Definition (protocol Specific)	9
Command Definition (coproc Specific)	10
EEPROM Values	11
Dios Coproc Basic Dios Library	13
Dios CPIO Dios Library	14
Links	16

Assembly

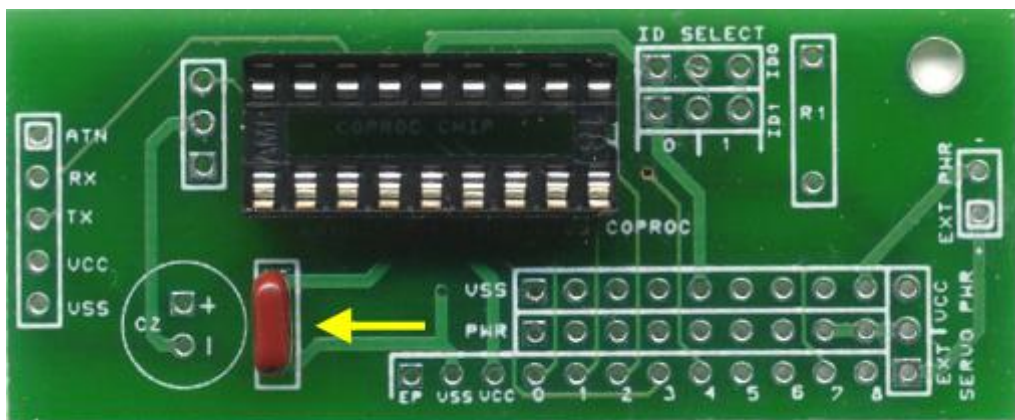
Each kit will take you about 30 minutes to build. You will need a soldering iron, solder, and wire cutters. Please read all the assembly instructions before beginning assembly. You can find a full color version of this document online at: <http://www.kronosrobotics.com/products/pdfs/pdfs.htm>



Step 1

Insert the 18 pin socket into the location marked U1. Carefully flip the board over so that the board rests on the socket. Make sure the notches are facing the right side of the board as shown.

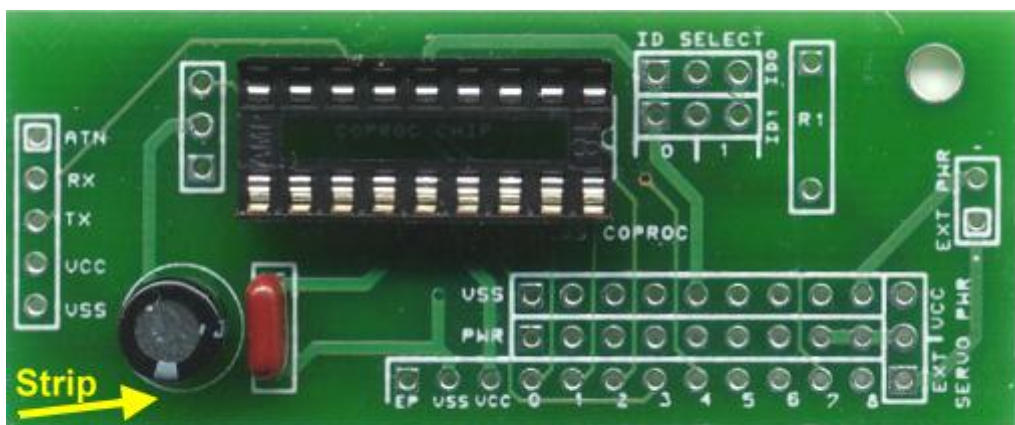
Solder in place.



Step 2

Insert the .1uf capacitor into the position labeled C1. (It's the dark red one)

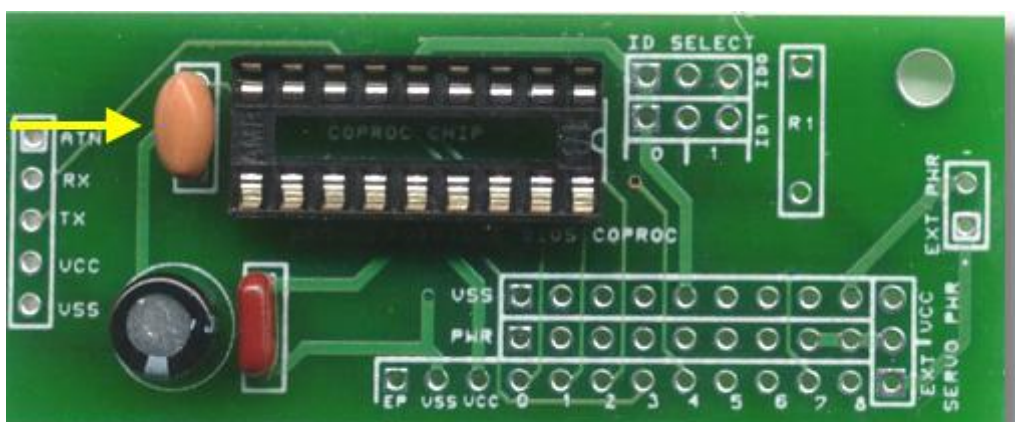
Solder in place and clip the excess leads.



Step 3

Insert the 100uf capacitor into the holes labeled C2. Make sure the negative side (strip) is facing the bottom of the board as shown.

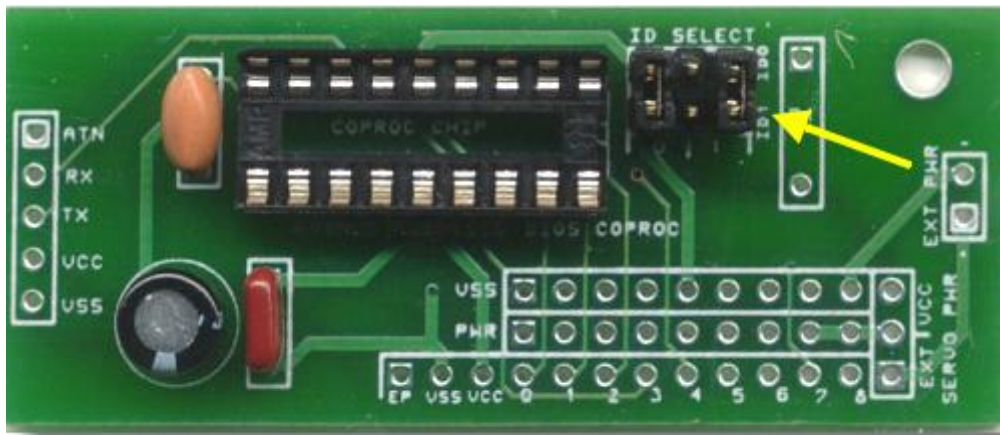
Solder in place and clip excess leads.



Step 4

Insert the 20Mhz resonator as shown.

Solder in place.



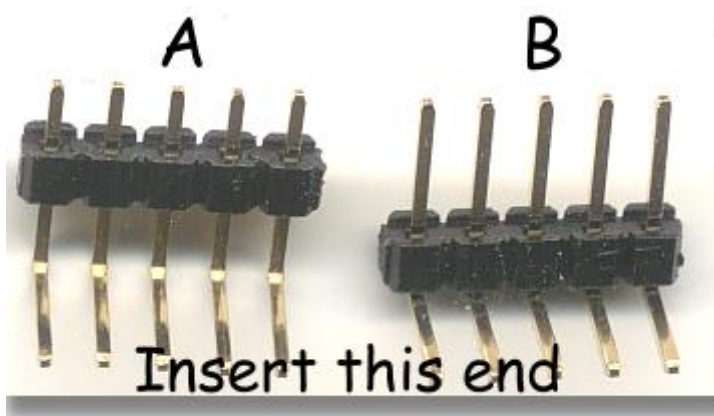
Step 5

Insert the 2, 3 pin headers as shown. Use a couple jumpers to hold in place.

Note the side with the small pins ends is inserted into the board from the top.

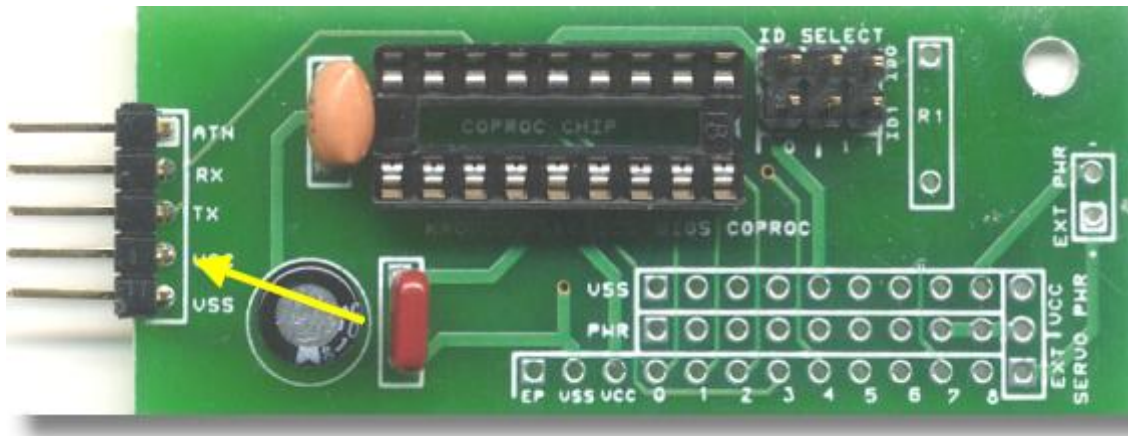
Solder in place.

Remove jumpers



Step 6

Take the 5 pin connector (A) and push the pins through as shown in B. This is optional. It all depends on how you want the buss connector to extrude from the board.

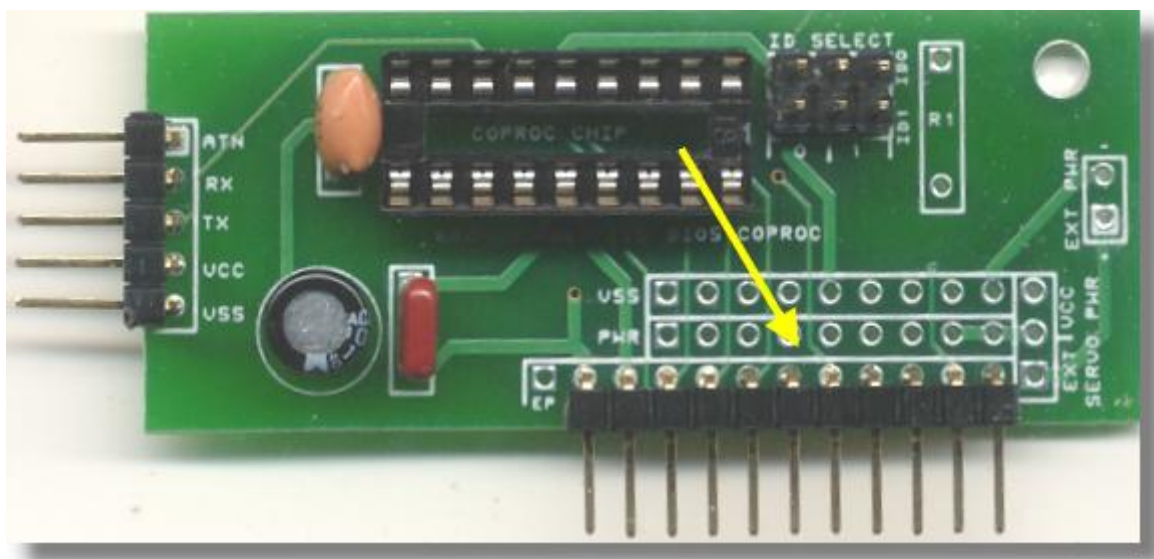


Step 7

Insert the 5 pin header as shown.

Solder in place

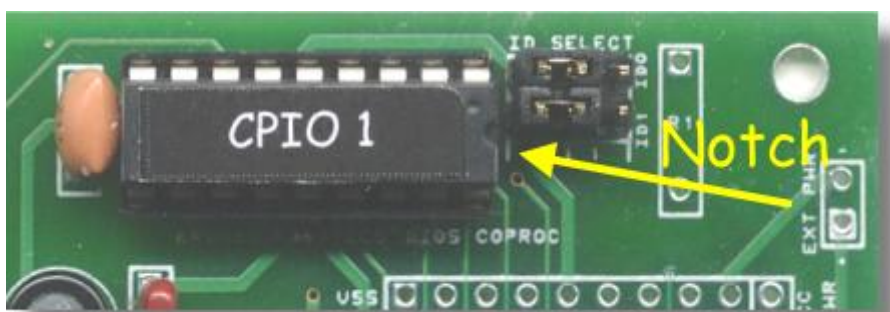
If you changed the pin lead length in step 6 you will need to clip the excess leads on the back of the board.



Step 8

If you wish modify the 11 pin header like you did the 5 pin header in step 6.

Insert as shown and solder in place. Clip excess leads on the bottom of board.



Step 9

Insert the CPIO1 chip as shown. Make sure the notch in the chip is facing right. If the sticker is not already on the chip place it there now.

Note

R1 is not used. Do not solder the included 10k resistor into this slot.

Hookup

The coprocs are identified by two parameters, Type and ID. The Type on this coproc is 1. The ID can be changed by setting the ID jumpers as shown below.



ID 0



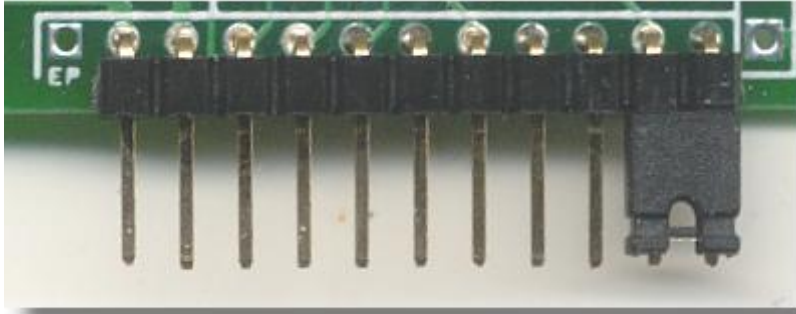
ID 1



ID 2



ID 3



The default speed of the Servo Coproc is 115200 baud. This can be changed via software.

You can change the baud speed at any time but the new speed will not take affect until the coproc has been reset. (power off then on)

If at any time you loose track of what baud rate you set the coproc or you have a device that cant communicate at the set speed you may force the coproc to a speed of 9600. You do this by powering down the coproc and inserting a jumper across ports 7 and 8 as shown. When you power the coproc the unit will be forced to 9600 baud. The coproc will stay at 9600 until the jumper is removed and the coproc has been reset. Once reset it resume whatever baud speed you have selected with the eewrite command.

Note that while in the forced baud speed mode servo ports 7 and 8 have will not be active.

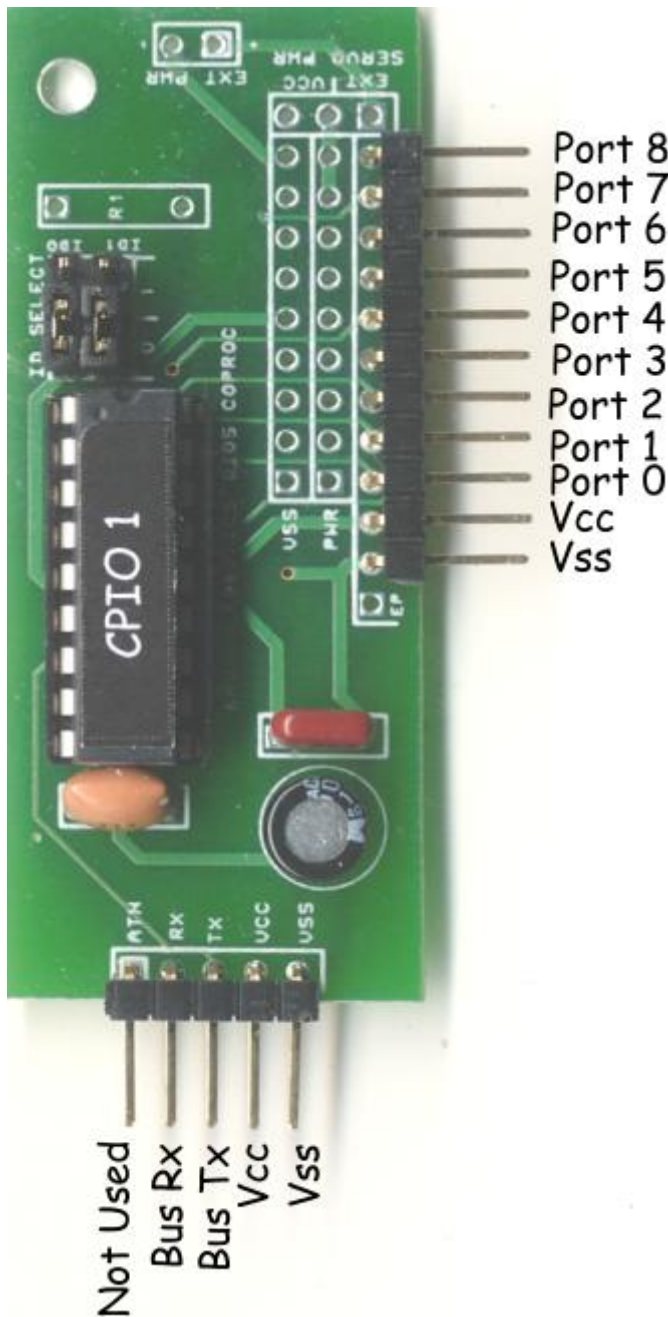
Hookup Notes

Coprocs can be connected to a common 5 pin bus or directly to the Dios or other microprocessor. If not connected to the Kronos Robotics minibus the **transmit bus lead must be held high with a 10k resistor**. This need only be done if you wish to read data from the coprocessor. Note that this is not the resistor marked R1 on the Coproc. This resistor must be connected externally on the connector. IE tie a 10K resistor (included) between the Tx lead and the VCC lead. (This is an external connection only. Don't use R1)

The Kronos Robotics minibus already has holding resistors on board.



Hookup



If connecting as a standalone processor you will have to hold the bus Tx lead high with a 10k resistor.

Port 8 Just connect the resistor between Vcc and bus Tx on the header. If connecting multiple coprocs the resistor should only be placed on one of them.

Port 7

Port 6

Port 5 If you are using the minibus the resistors are already located on the bus.

Port 4

Port 3

Port 2

Port 1

Port 0

Vcc

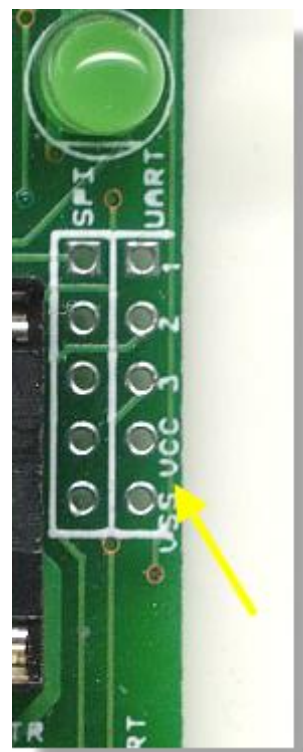
Vss

When connecting to the Dios make the following connections:

Bus Rx—IO port 8
 Bus Tx—IO port 9
 Vcc—Dios Vcc
 Vss—Dios Vss

The Dios Ultra has a small header pad located on the right side next to the power LED. With the minibus kit a small header can be placed here to allow quick connections to the coproc bus.

You can also place your own male or female header here if you like.



Protocol Definition

The coprocs use a variable length packet. It can be anywhere from 2 to 9 bytes in length. All coprocs on the bus monitor and store the packet. Once all bytes from a packet are received each coproc on the bus will check the sent Type and ID against its own Type and ID. If there is no match the coproc will discard the packet. If the Type and ID do match the coproc will pass the data to its command processor and act on the data sent to it. There is one exception to this rule.

Type 0 : Targets all Types and IDs (every thing on bus will act on data)

Byte Definition

Byte 0

Bits 0-1 ID

Bits 2-7 Type

We refer to the Type and ID as an address pair. You will see it represented as (Type:ID) or (x:x). At times I will refer to this pair as AP (address pair)

Byte 1

Bits 0-4 Cmd

Bits 5-7 Packet Length

These Cmd and packet length are closely related so many times they will be referred to as command pair or just command.

Bytes 2—8

The actual number of bytes needed will depend on the command sent. IE its packet length.

IE

Byte2 Length = 1

Byte3 Length = 2

Byte4 Length = 3

Byte5 Length = 4

Byte6 Length = 5

Byte7 Length = 6

Byte8 Length = 7

Examples

0,0 Tells all coprocs connected to turn off transmitters

5,129,99,200,30,15 Tells Coproc (1:1) to write value 15 to address 30 of its eeprom

5,34,30 Tells coproc (1:1) to return the value of address location 30 of its eeprom

Lets break this one down

Byte 0 (5)

00000**101** ID = 1

00000101 Type = 1 or (4 in actual byte. Its shifted 2 bits)

Byte 1 (34)

100**00010** Cmd = 2

10000010 Packet Length = 4 (128 in actual byte. Its shifted 5 bits. Add 4 more bytes to packet)

Byte2 (30)

Address to read

Command Definition

Protocol Specific Commands

Cmd 0

AP,0
Turn off Transmitter

Cmd 1 (129)

AP,129,200,99,addr,data
Write data to internal eeprom.

Byte 2 must be 99
Byte 3 must be 200
Byte 4 Eeprom address to write to
Byte 5 Data to write

There are 128 eeprom slots available. Slots 0-9 are reserved for protocol specific parameters such as baud rate and internal timing.

The remaining slots are coprocessor specific.

You must have a small pause after writing to a eeprom address. IE the buss must be quiet for 50ms after a write as all internal interrupts are stopped during the write.

Cmd 2 (34)

AP,34,addr
Requests a the coproc to transmit the contents of eeprom address.

Cmd 3 (3)

AP,3
Tells the coproc to load its default values from eeprom.

Cmd 4 (4)

AP,4
Request version number from coproc.

Cmd 5 (5)

AP,5
Request status from coproc.

Coproc Specific commands

Readports

Cmd 8 (40)

AP,40,data

Writes to ports 0-7 with one command. Bits 0-7 in data map out to the ports 0-7. If bit is 1 then port will be high. If bit is 0 then port will be low.

Writeports

Cmd 9 (9)

AP,9

Request version status of ports 0-7 with one command. Data is returned with one byte. With ports 0-7 mapped to bits 0-7

Setiostate

Cmd 10 (74)

AP,74,port,state

Sets the state of an output port. If state is 0 then port will be low. If 1 then port will be high

Setiodir

Cmd 11 (75)

AP,75,port,dir

Sets the direction of the port. If dir is 0 then port is output. If 1 then port will be set to input.

Getiostate

Cmd 12 (44)

AP,44,port

Returns the state of the passed port. A value of 0 or 1 will be received.

Setpwmflag

Cmd 13 (77)

AP,77,port,flag

Turns on the software pwm for the given port. If 0 pwm is off. If 1 then pwm is on.

Setpwmduty

Cmd 14 (78)

AP,78,port,duty

Sets the duty for software pwm port. Valid values are 0-maxcount

Setmaxcount

Cmd 15 (47)

AP,47,maxcount

Sets the maxcount of the pwm generator. Default is 255. If changed to lower values the frequency of the software pwm can be increase at a reduced duty resolution. 0-255

SetHWpwmstate

Cmd 16 (48)

AP,48,state

Turns on or off the hardware pwm port. If 0 then off. If 1 then on. Hardware pwm port is located on port 4.

SetHWpwmperiod

Cmd 17 (49)

AP,49,period

Sets the period/frequency of the hardware pwm port. 0-255. The lower the number the higher the frequency.

SetHWpwmduty

Cmd 18 (50)

AP,50,duty

Sets the duty of the hardware pwm port. Match this number to the period. For 100% duty cycle.

SwtHWpwmcourse

Cmd 19 (51)

AP,51,course

Sets the range of the hardware pwm port. Only values of 0-2 is allowed. 0=high range, 1=mid range, 2=low range

SwtHWpwmfineduty

Cmd 20 (51)

AP,51,course

You can fine tune the duty cycle with this setting. Only a value of 0-2 is allowed.

EEPROM Values

(Protocol Specific EEPROM Values)

Byte 0

Used for testing should be a value of 255

Byte 1

Version

Byte 2

Type

Byte 3

Reserved

Byte 4

Idlecount.

When no packet data is received an internal packet counter is incremented. When that counter overflows from 255 to 0 the packet byte count is automatically reset to 0. It is an auto packet re-sync. In other words if you stop transmitting packets to the coproc bus all the coprocs will auto re-sync. The Idlecount value is the starting value for this counter. The default value is 254. (very fast). Normally this should never have to be changed as there are quite a few error recovery routines built into the coprocs.

Byte5

Txwaithigh

Byte6

Txwaitlow

These bytes make up a transmit delay for sending requested bytes. When a command is received that requests information a small delay is inserted. This delay is in microsecond and the default is 1us. If you are using a slower processor you may need to change this value

Byte7

Boot baud index

Default is 10 (115200 baud)

Valid Values

129=9600

64=19200

32=38400

21=57600

10=115200

4=250000

1=625000

0=1250000

Byte8

TxIdlecount

When no packet data is received or transmitted an internal packet counter is incremented. When that counter overflows from 255 to 0 the transmitter will be taken off line. The TxIdlecount is the starting value for this counter.

The default is 254 (very fast)

Byte 9

Reserved

(Coproc Specific EEprom Values)

Byte 10

Startup port direction for ports 0-7. Bits 0-7 map to ports 0-7. If 1 then port set to input. If 0 then port set to output.

Byte 11

Startup port direction for port 8. Bit 0 map to port 8. If 1 then port set to input. If 0 then port set to output.

Byte 12

Startup port state for ports 0-7. Bits 0-7 map to ports 0-7. Only valid if port dir is output. If 1 then port is high. If 0 then port is low.

Byte 13

Startup port state for port 8. Bit 0 maps to ports 8. Only valid if port dir is output. If 1 then port is high. If 0 then port is low.

Byte 14

Startup countmax for software pwm.

Byte 15

Startup flags for pwm ports. Bits 0-7 map to ports 0-7. If bit is high then pwm is active. If 0 then pwm is not active. Note that dir must be output for pwm to be active as well.

Byte 16

Startup flags for pwm port 8. Bits 0 map to ports 8. If bit is high then pwm is active. If 0 then pwm is not active. Note that dir must be output for pwm to be active as well.

Bytes 17–25

Startup duty cycle for each port when in software pwm mode.

Byte 18

Hardware PWM period startup

Byte 19

Hardware PWM duty startup

Byte 20

Hardware PWM fine duty startup

Byte 21

Hardware PWM course startup

Byte 22

Hardware PWM stat startup. When 0 then hardware pwm disabled. When 1 then hardware PWM will be enabled at startup.

Dios Coproc Basic Library

CPinit(*baud*)

This function starts the CP engine

baud This optional command sets the baud rate. If omitted a baud rate of 115200 will be used.

CPversion(*type, id*)

This function returns the current version of the given coproc.

type The target coproc type 1-63. If 0 is used all types on buss are the target.

id The target coproc id 0-3. If type is 0 then this field is ignored.

CPtxoff(*type, id*)

This function turns off the target coprocs transmitter.

type The target coproc type 1-63. If 0 is used all types on buss are the target.

id The target coproc id 0-3. If type is 0 then this field is ignored.

CPloaddefaults(*type, id*)

The target coproc loads its eeprom defaults.

type The target coproc type 1-63. If 0 is used all types on buss are the target.

id The target coproc id 0-3. If type is 0 then this field is ignored.

CPwriteEEProm(*type, id, address, value*)

Write a value to the eeprom of the target coproc.

type The target coproc type 1-63. If 0 is used all types on buss are the target.

id The target coproc id 0-3. If type is 0 then this field is ignored.

address The address of the eeprom to write to. 0-128

value The value to write. 0-255

CPreadEEProm(*type, id, address*)

Reads the value of an address point of the target eeprom.

type The target coproc type 1-63. If 0 is used all types on buss are the target.

id The target coproc id 0-3. If type is 0 then this field is ignored.

address The address to read.

CPsendbytes(*type, id, data, data,*)

Sends up to 9 bytes to the target coproc.

type The target coproc type 1-63. If 0 is used all types on buss are the target.

id The target coproc id 0-3. If type is 0 then this field is ignored.

data These are the data bytes to send. Just include the bytes needed the proper protocol values will be set for you.

CPgetstatus(*type, id*)

Returns coproc status.

There are 8 indicators. One for each bit. Only bit 7 is common to all coprocs. If bit 7 is high then the coproc has initiated the protocol recovery routine. This can happen for various reasons but mostly if data is sent too fast.

type The target coproc type 1-63. If 0 is used all types on buss are the target.

id The target coproc id 0-3. If type is 0 then this field is ignored.

Dios CPIO Library

CPinit(*baud*)

This function starts the CP engine

baud This optional command sets the baud rate. If omitted a baud rate of 115200 will be used.

IOPORTS

CPiostate(*id, port, state*)

This function sets the state a given port.

id The target coproc id 0-3. If type is 0 then this field is ignored.

port The IOport 0-8 you wish to change.

state The stat 0/1 of the IOport you want to set. If 0 then the port is set low. If 1 then the port will be set high. Note that you can use **CPhigh** and **CPlow** constants.

CPiogetstate(*id, port*)

This function returns the state of the given port. Will return 0 or 1.

id The target coproc id 0-3. If type is 0 then this field is ignored.

port The IOport 0-8 you wish to change.

CPiodir(*id, port, dir*)

This function is used to set the IOport as input or output.

id The target coproc id 0-3. If type is 0 then this field is ignored.

port The IOport 0-8 you wish to change.

dir The direction to set the port. 1 will set the port to input. 0 will set the port to output. Note that you can use the **CPinput** and **CPoutput** constantans.

CPioread(*id*)

Returns the state of ports 0-7 in one pass. Each bit will represent the state of that port.

id The target coproc id 0-3. If type is 0 then this field is ignored.

CPiowrite(*id, dat*)

Sets the state of IO ports 0-7 in one pass. Each bit in dat represents the state you want to set.

id The target coproc id 0-3. If type is 0 then this field is ignored.

dat The data used to set ports 0-7. Note that only bits 0-7 are used.

Hardware PWM

CPioHWpwminit(*id*)

Starts the hardware PWM generator on port 4

id The target coproc id 0-3. If type is 0 then this field is ignored.

CPiostopHWpwm(*id*)

Turns off the hardware PWM generator.

id The target coproc id 0-3. If type is 0 then this field is ignored.

Dios CPIO Library cont

Hardware PWM cont

CPioHWpwmPeriod(id, period)

Sets the hardware pwm period/frequency.

id The target coproc id 0-3. If type is 0 then this field is ignored.

period This value sets the frequency. A value of 0-255 is valid.

CPioHWpwmDuty(id, duty)

Sets the duty of the hardware PWM port.

id The target coproc id 0-3. If type is 0 then this field is ignored.

duty This value is used to set the duty cycle. if period is set to 255 then a value of duty of 128 will give you a duty cycle of 50%. A duty of 230 will give you a 90% duty cycle.

CPioHWpwmCourse(id, value)

Sets the range of the PWM period

id The target coproc id 0-3. If type is 0 then this field is ignored.

value Only a range of 0-2 are allowed. 0 the fastest range and 2 the slowest.

CPioHWpwmfduty(id, value)

This function will allow you to fine tune the duty cycle.

id The target coproc id 0-3. If type is 0 then this field is ignored.

value Only a range of 0-2 are allowed.

Software PWM

CPioPWMmaxcount(id, maxcount)

By changing this value you can change the frequency of the software pwm generator. However the faster the frequency the less resolution.

id The target coproc id 0-3. If type is 0 then this field is ignored.

maxcount Sets the maximum count. A value of 0-255 is valid.

CPioPWMflag(id, port, state)

Allows you to turn on the software pwm generator for each port.

id The target coproc id 0-3. If type is 0 then this field is ignored.

port Sets the maximum count. A value of 0-255 is valid.

state A value of 1 turns on the software pwm for the port given.

CPioPWMduty(id, port, duty)

Allows you to set the duty cycle for each port for the software pwm generators.

id The target coproc id 0-3. If type is 0 then this field is ignored.

port Sets the maximum count. A value of 0-255 is valid.

duty Sets the duty for the given port. 0-maxcount

All Coproc libraries are always included with the Dios Software. Dios libraries always include help files.

Links

Be sure to visit the Kronos Robotics web site for more information and updates and example code.

Web Site

<http://www.kronosrobotics.com>

Full color assembly instructions

<http://www.kronosrobotics.com/products/pdfs/pdfs.htm>