Serial Servo Controller – BM013_SERVO

OPEN SOURCE HARDWARE MODULE

hardware made easy
Solutions Cubed
designservices@solutions-cubed.com
phone – 530.891.8045
256 East First Street
Chico, CA 95928

Contact Solutions Cubed, LLC for your custom designs:

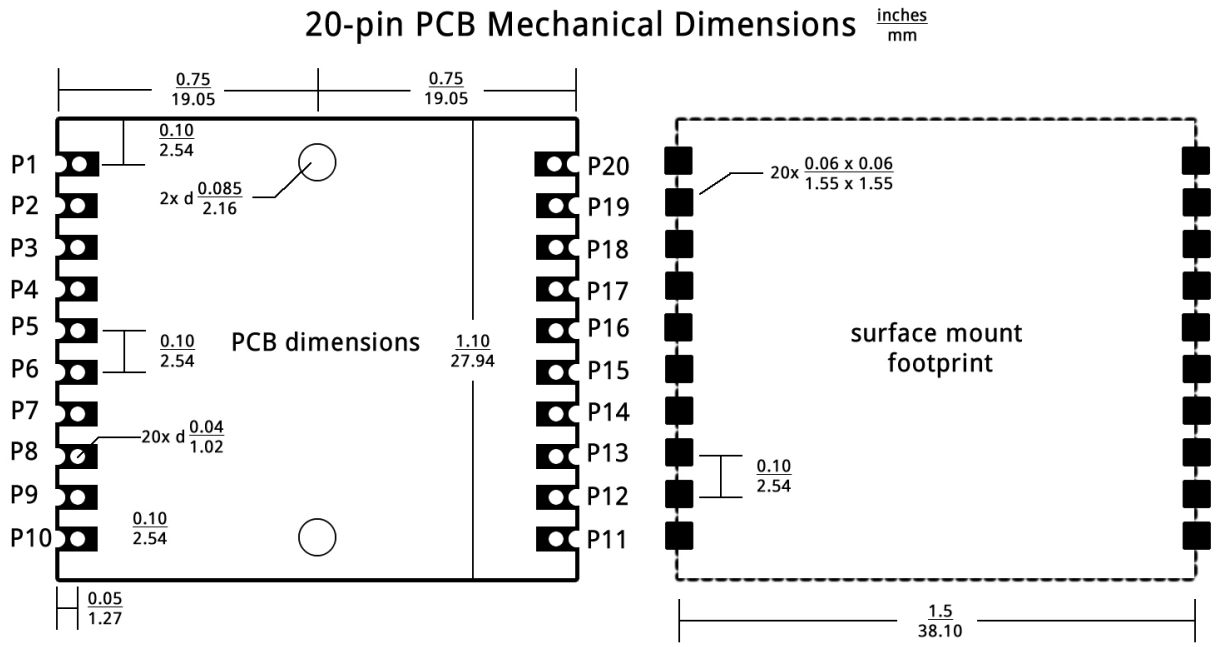
Solutions Cubed is an innovative electronic design firm. We have created successful designs for a myriad of industries including mass produced consumer products, deep-sea robotic components, and encrypted encoders for the banking industry. We love meeting new customers and are interested in hearing about your design needs.

Product Description:

This module can be used to generate pulses to control up to 10 servos. Pulse width can be controlled with 1uS resolution. A logic-level serial communication interface is the primary means of controlling the module.

- Simple serial data interface at 9600BPS for control
- 3.3V or 5.0V operation
- Control pulse width generation with 1uS resolution for 10 servos
- Set maximum and minimum allowable pulse width
- Adjustable rate-of-change for servo pulse width down to 1uS per pulse period
- 3 general purpose input/output pins are available
- All servo pulses can be disabled by setting a specific register bit
- Open-source firmware and hardware so you can modify the design as you like

Dimensions:



Specifications:

Characteristic	Min	Typ	Max	Unit	Notes
VDD Operating voltage	3		5.25	V	VDD pins
Operating current		7		mA	No motors/servos attached
VIN Voltage	6		24	V	Operating near the maximum voltage may require external protection from voltage spikes
Servo pulse output resolution		1		uS	
Servo pulse output range	0		25000	uS	
Servo pulse output accuracy		+/-20		uS	Servo pulses are generally longer by 5-10uS due to interrupt subroutine processing time. This error is consistent and can be accommodated for by sending shorter pulse widths making accuracy of pulse output much better than +/-20uS.
Servo pulse output period	12	24	510	mS	this is user adjustable but defaults to 24mS
Operating temperature	-40		+85	°C	
Servo output pin current sourcing			15mA		have series 270Ω resistor on board
GPIx pin current sourcing			10mA		have series 270Ω resistor on board and tied to LED

Pin Functions and Notes

#	Name	Maximum Voltage	Notes
1	DOUT	VDD	has series 270Ω resistor on board
2	DIN	VDD	has series 270Ω resistor on board
3	GPIO0	VDD	has series 270Ω resistor on board, controlled via MODE register
4	GPIO1	VDD	has series 270Ω resistor on board, controlled via MODE register
5	GPIO2	VDD	has series 270Ω resistor on board, controlled via MODE register
6	GND	N/A	Ground return
7	3.3V	3.3V output	3.3V power supplied by voltage at VIN, can source 50mA
8	VDD	5.25V	Microcontroller power supply, connect to 5V, 3.3V, or some external supply voltage
9	5V	5V output	5V power supplied by voltage at VIN, can source 50mA
10	VIN	24V	Voltage input to on-board 3.3V and 5V power supplies
11	S_OUT1	N/A	has series 270Ω resistor on board
12	S_OUT2	N/A	has series 270Ω resistor on board
13	S_OUT3	N/A	has series 270Ω resistor on board
14	S_OUT4	N/A	has series 270Ω resistor on board
15	S_OUT5	N/A	has series 270Ω resistor on board
16	S_OUT6	N/A	has series 270Ω resistor on board
17	S_OUT7	N/A	has series 270Ω resistor on board
18	S_OUT8	N/A	has series 270Ω resistor on board
19	S_OUT9	N/A	has series 270Ω resistor on board
20	S_OUT10	N/A	has series 270Ω resistor on board

Programming Header J1

#	Name	Maximum Voltage	Notes
1	MCLR	VDD	Reset line for in-circuit programmer
2	5V	VDD	connects to VDD
3	GND	N/A	ground return
4	PGD	VDD	In-circuit programming data line
5	PGC	VDD	In-circuit programming clock line

User Notes/Tips

1. RC servos can draw large amounts of current. It is best to wire the servo power lines directly into your battery. See application schematics for examples.

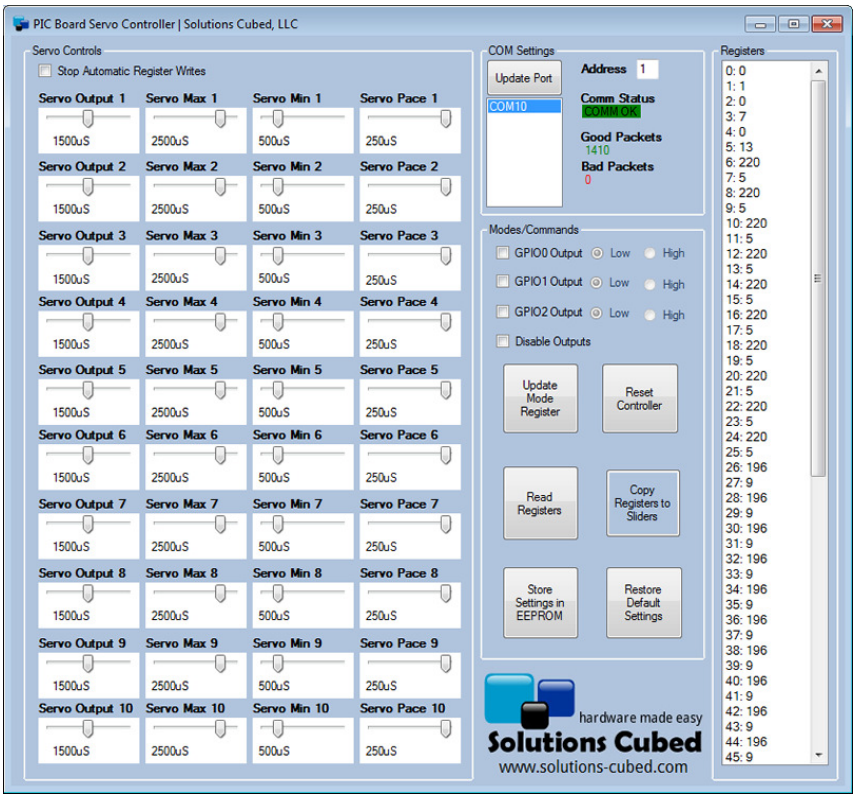
2. The PIC16F1829 firmware is open-source and may be programmed using a free development environment and C compiler from Microchip. We used MPLAB X and XC8. The module may be debugged and programmed with the low cost devices such as Microchip's PICKit3. This will connect to J1 on the module by using our CS005 Programming Adapter kit available at www.solutions-cubed.com.

3. Servo pulse output duration is typically longer than the value you send by 5-10uS. This relates to the time it takes for the interrupt routines to process code, and is a consistent error. We did not adjust for this in the firmware. If you want more accurately timed servo output pulses you can just send shorter pulse values.

4. Servo pulses are generated sequentially. S_OUT1, then S_OUT2, etc.

5. If the user sets the servo output pulses so the sum of them exceed the period of the servo update rate (default of 26mS), then Servo_Period register should be set exceed the sum of the pulses. Example, if all ten outputs can be as long as 3mS each the Servo_Period should be set to 16 (32mS between).

6. Test Software is available to test the serial interface and various control settings of this module. The test software source code is available on our web site.



Communication Protocol

Hardware Usage:
The asynchronous serial communication takes place at 9600BPS using eight data bits, no parity, and one stop bit (9600, n, 8, 1). Data is sent LSB first. There is no hardware flow control. Serial data is received by the module on its DIN pin. It sends data on its DOUT pin.

Communication Sequence Overview:
The serial protocol is a master/slave interface. This module is the slave device and unless otherwise noted does not initiate communication.

There are four types of communication packets. They are Read Packets, Write Packet, ACK Packets, and Reply Packets. The master always initiates a communication sequence with a command and a slave always terminates the communications sequence with a reply or ACK.

Packet Byte	Representation
Operation	Read = 209, Write = 210
Address	default is 1
Length	3 to 30
Packet Data	Message specific
Checksum	sum of all characters in packet proceeding the checksum

Figure 5.2: Packet information

Operation: There are two operations: read or write. A read operation allows the master to specify which registers to read from the module. A write operation allows the master to write to specific registers.

Address: The default device address is 1. This value may be changed using a write operation.

Length: This byte tells the receiver how many bytes are *following* the length byte. This includes the checksum.

Packet Data: The specific data that is necessary for a complete message.

Checksum: A simple sum of *all* of the bytes in the packet before the checksum. For example, if the packet data is 209, 1, 3, 0, 1, the checksum would be 214.

Since the checksum is a single byte value only the remainder is sent for values greater than 255. If the sum of a packet's bytes is 257, the checksum would be 1 (257/256 = 1 with a remainder of 1). If the sum is 512 the checksum is 0 (512/256 = 2 with a remainder of 0).

In most cases when you are summing values in a byte sized variable any overflow is lost, and the remainder is all that is left. This is the checksum.

Read Operation
A read operation allows the master to read specific registers from the module.

Operation Byte	Device Address	Length	Start Index	Number of Registers	Checksum

Figure: Form of a read command

The "Start Index" is the register index value that the read should begin from. For example, if you want to start your read at register 4, then the Start Index is 4. The "Number of Registers" is number of registers to read, including the register identified by the Start Index. The example below shows a read command of register 4. A reply packet will be returned when the read operation is accepted.

Master Sends: 209, 1, 3, 4, 1, 218
Slave returns: reply packet

Write Operation
A write operation allows the master to write specific registers to the module.

Operation Byte	Device Address	Length	Start Index	Data1	Data N	Checksum

Figure: Form of a write command

The "Start Index" identifies the first register that will be written to. The "Data1" register is the data to write to the Start Index register. With the Write command it is possible to write to more than one register at a time. The data will be written to sequential register starting at the Start Index. In the figure above the "Data N" byte will be written to the "Start Index" +1 register. The example below shows a write of 150 to register 11 and 151 to register 12.

Master Sends: 210, 1, 4, 11, 150, 151, 15
Slave returns: ACK packet

You'll notice that the sum of the bytes in the packet is 527. Since the checksum is a byte sized value only the overflow is sent (527 = hexadecimal 0x20F, therefore the lower byte is sent, 0x0F, or decimal 15.

Reply Packet
The module sends Reply Packets to the Master in response to a Read Operation. Each Reply Packet will begin with the slave address (default to "1"). Next is a length byte, followed by the message data. The last byte in the packet is the checksum.

Device Address	Length	Data1	DataN	Checksum

Figure: Reply packet representation

The "Data" in the reply packet is the register data that the master requested using a read command.

ACK Packet
The module always sends an ACK packet to the Master in response to a Write Operation. Each ACK Packet is a single byte.

ACK
6

Figure: ACK packet representation

The module sends the ACK packet *after* the requested write is completed. If the master does not receive an ACK after an appropriate period of time, it may assume that the write operation did not work.

Error Detection:
Error detection is accomplished by inspection of the received data and making sure that the data was received in a timely and appropriate fashion. Inspection of the data packets will be performed by…

- verifying that all elements of the packet are present
- making sure that the message is the correct length
- verifying the checksum
- verifying that the message is supported by the Slave
- testing all values with limited range
- inter-character time

The inter-character time is the time between successive characters (bytes) in the same packet. The maximum time allowed is 2ms.

Serial Communication Examples:

Example 1 - Setting S_OUT1 to 2000uS (registers 6, 7). 2000 decimal is hexadecimal '07D0'; we send the lower byte 'D0' (decimal 208) to register 6 and the upper byte '07' (decimal 7) to register 7. Setting the index of the register we are writing to at 6 will cause the first data byte to be written there, and the following data byte will go into register 7.

Master Sends: 210, 1, 4, 6, 208, 7, 180
Slave returns: 6

Example 2 – Reading the Indicator register whose contents are 0.

Master Sends: 209, 1, 3, 4, 1, 218
Slave returns: 1, 2, 0, 3

Register Definitions:

This module is controlled by reading from and writing to internal registers via the previously described serial interface. The registers are defined below.

Index	Register	Range	Default Value	Description
0	Firmware	1-255	NA	Firmware revision loaded into the module.
1	Address	0-255	1	The serial address this unit will respond to.
2	Command	0-255	0	Commands specific to the module may be implemented by writing to this register. See the detailed register definition.
3	Mode	0-255	7	Operating modes specific to the module may be implemented by writing to this register. See the detailed register definition.
4	Indicator	0-255	NA	Indicators specific to this module may be read from this register. See the detailed register definition.
5	Servo_Period	0-255	13	The period of time before a servo pulse repeats in 2mS intervals. Period is 2mS*Servo_Period. Therefore, the default value is 26mS.
6	Servo_Lo_Out1	0-255	220	Low byte of the length of the pulse output at the associated S_OUTx pin (in uS).
7	Servo_Hi_Out1	0-255	5	High byte of the length of the pulse output at the associated S_OUTx pin (in uS). Example, pulse width at S_OUT1 = Servo_Hi_Out1*256 + Servo_Lo_Out1. Default is 1500uS.
8	Servo_Lo_Out2	0-255	220	see above
9	Servo_Hi_Out2	0-255	5	see above
10	Servo_Lo_Out3	0-255	220	see above
11	Servo_Hi_Out3	0-255	5	see above
12	Servo_Lo_Out4	0-255	220	see above
13	Servo_Hi_Out4	0-255	5	see above
14	Servo_Lo_Out5	0-255	220	see above
15	Servo_Hi_Out5	0-255	5	see above
16	Servo_Lo_Out6	0-255	220	see above
17	Servo_Hi_Out6	0-255	5	see above
18	Servo_Lo_Out7	0-255	220	see above
19	Servo_Hi_Out7	0-255	5	see above
20	Servo_Lo_Out8	0-255	220	see above
21	Servo_Hi_Out8	0-255	5	see above
22	Servo_Lo_Out9	0-255	220	see above
23	Servo_Hi_Out9	0-255	5	see above
24	Servo_Lo_Out10	0-255	220	see above
25	Servo_Hi_Out10	0-255	5	see above

Register Definitions (continued):

Index	Register	Range	Default Value	Description
26	Servo_Lo_Max1	0-255	196	Low byte of the maximum pulse output allowed at the associated S_OUTx pin (in uS).
27	Servo_Hi_Max1	0-255	9	High byte of the maximum pulse output allowed at the associated S_OUTx pin (in uS). Example, pulse width at S_OUT1 <= Servo_Hi_Max1*256 + Servo_Lo_Max1. Default is 2500uS.
28	Servo_Lo_Max2	0-255	196	see above
29	Servo_Hi_Max2	0-255	9	see above
30	Servo_Lo_Max3	0-255	196	see above
31	Servo_Hi_Max3	0-255	9	see above
32	Servo_Lo_Max4	0-255	196	see above
33	Servo_Hi_Max4	0-255	9	see above
34	Servo_Lo_Max5	0-255	196	see above
35	Servo_Hi_Max5	0-255	9	see above
36	Servo_Lo_Max6	0-255	196	see above
37	Servo_Hi_Max6	0-255	9	see above
38	Servo_Lo_Max7	0-255	196	see above
39	Servo_Hi_Max7	0-255	9	see above
40	Servo_Lo_Max8	0-255	196	see above
41	Servo_Hi_Max8	0-255	9	see above
42	Servo_Lo_Max9	0-255	196	see above
43	Servo_Hi_Max9	0-255	9	see above
44	Servo_Lo_Max10	0-255	196	see above
45	Servo_Hi_Max10	0-255	9	see above
46	Servo_Lo_Min1	0-255	244	Low byte of the minimum pulse output allowed at the associated S_OUTx pin (in uS).
47	Servo_Hi_Min1	0-255	1	High byte of the minimum pulse output allowed at the associated S_OUTx pin (in uS). Example, pulse width at S_OUT1 >= Servo_Hi_Min1*256 + Servo_Lo_Min1. Default is 500uS.
48	Servo_Lo_Min2	0-255	244	see above
49	Servo_Hi_Min2	0-255	1	see above
50	Servo_Lo_Min3	0-255	244	see above
51	Servo_Hi_Min3	0-255	1	see above
52	Servo_Lo_Min4	0-255	244	see above
53	Servo_Hi_Min4	0-255	1	see above
54	Servo_Lo_Min5	0-255	244	see above
55	Servo_Hi_Min5	0-255	1	see above

Register Definitions (continued):

Index	Register	Range	Default Value	Description
56	Servo_Lo_Min6	0-255	244	see above
57	Servo_Hi_Min6	0-255	1	see above
58	Servo_Lo_Min7	0-255	244	see above
59	Servo_Hi_Min7	0-255	1	see above
60	Servo_Lo_Min8	0-255	244	see above
61	Servo_Hi_Min8	0-255	1	see above
62	Servo_Lo_Min9	0-255	244	see above
63	Servo_Hi_Min9	0-255	1	see above
64	Servo_Lo_Min10	0-255	244	see above
65	Servo_Hi_Min10	0-255	1	see above
66	Servo_Pace1	0-255	250	The pace register sets the allowable rate of change for the associated servo pulse output. The units are uS per pulse period. Example, setting the value to 1 forces the pulse output to change no more than 1uS per Pulse_Period. Therefore, an output changing from 1000uS to 1500uS would take 500 pulse periods if the pace register is set to 1 (13 seconds at the default pulse period). It would take 4 pulse periods at the default of 250uS. Setting this register to 0 disables the pacing function.
67	Servo_Pace2	0-255	250	see above
68	Servo_Pace3	0-255	250	see above
69	Servo_Pace4	0-255	250	see above
70	Servo_Pace5	0-255	250	see above
71	Servo_Pace6	0-255	250	see above
72	Servo_Pace7	0-255	250	see above
73	Servo_Pace8	0-255	250	see above
74	Servo_Pace9	0-255	250	see above
75	Servo_Pace10	0-255	250	see above
76	Programmed	0-255	244	Low byte of the minimum pulse output allowed at the associated S_OUTx pin (in uS).

Detailed Register Descriptions:

Command Register (index = 2): Writing values to the Command register implements specific commands. Once these commands are executed the register value is returned to 0.

Command	Value	Description
Restore	1	Restores internal registers to their default settings and stores those values in EEPROM so they become the power on default values.
Store	2	Stores the current register values in EEPROM so they become the power on default values.
Reset Controller	3	Forces the module's microcontroller into a watchdog timer reset.

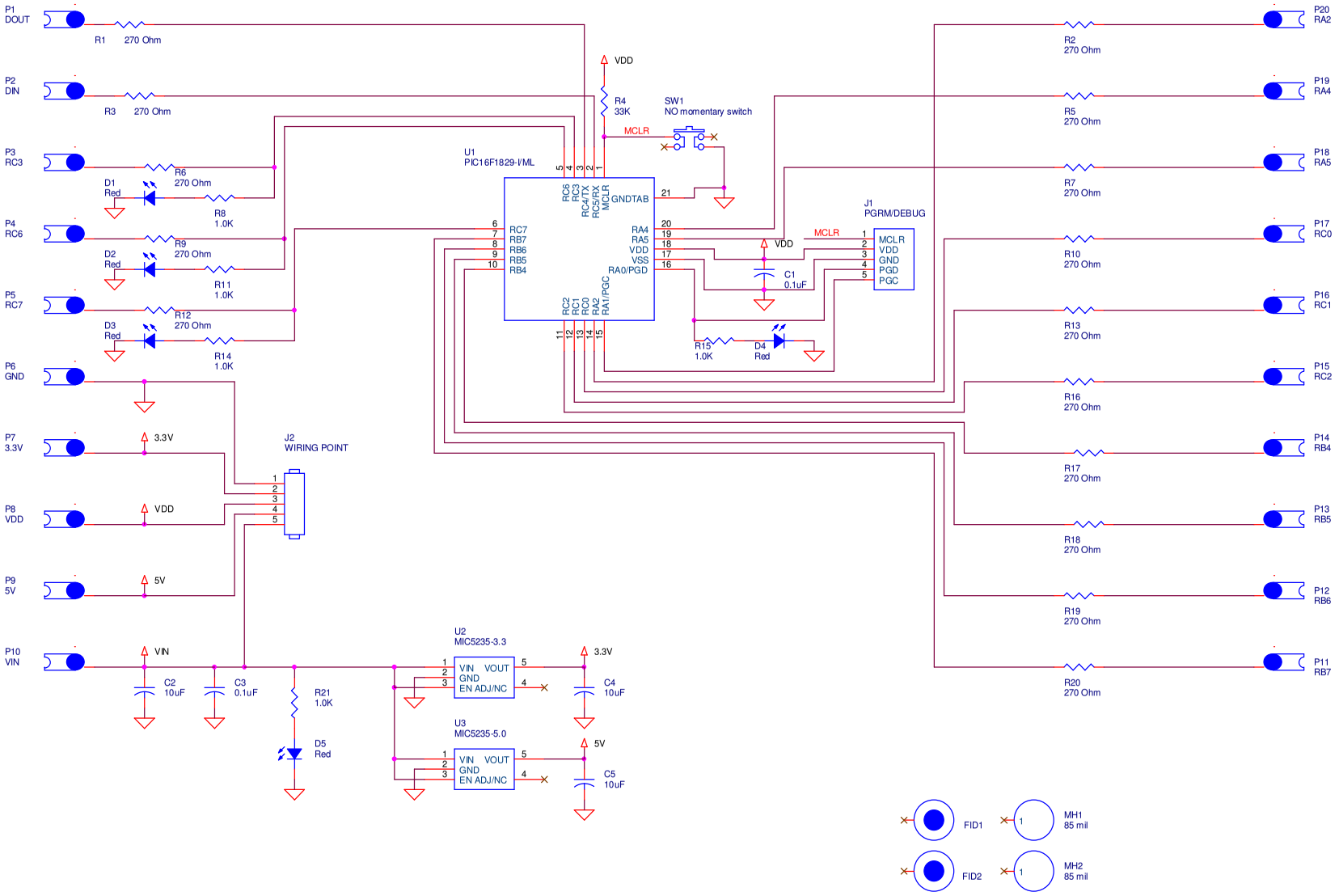
Mode Register (index = 3): Setting specific bits in the Mode register causes the module to implement special operating modes. A bit is set by writing a "1" to it. Multiple special modes can be implemented at the same time, although not all modes work together.

Mode	Bit	Description
GPIO0_Mode	0	1: Pin 3 is configured as an input. 0: Pin 3 is configured as an output, whose state is defined by Mode bit 4.
GPIO1_Mode	1	1: Pin 4 is configured as an input. 0: Pin 4 is configured as an output, whose state is defined by Mode bit 5.
GPIO2_Mode	2	1: Pin 5 is configured as an input. 0: Pin 5 is configured as an output, whose state is defined by Mode bit 6.
	3	unused
GPIO0_State	4	1: Pin 3 set to VDD. 0: Pin 3 is set to 0V.
GPIO1_State	5	1: Pin 4 set to VDD. 0: Pin 4 is set to 0V.
GPIO2_State	6	1: Pin 5 set to VDD. 0: Pin 5 is set to 0V.
Disable_Pulses	7	1: S_OUT1 through S_OUT10 are set to 0V 0: Pulses are present on SOUT1 through S_OUT10

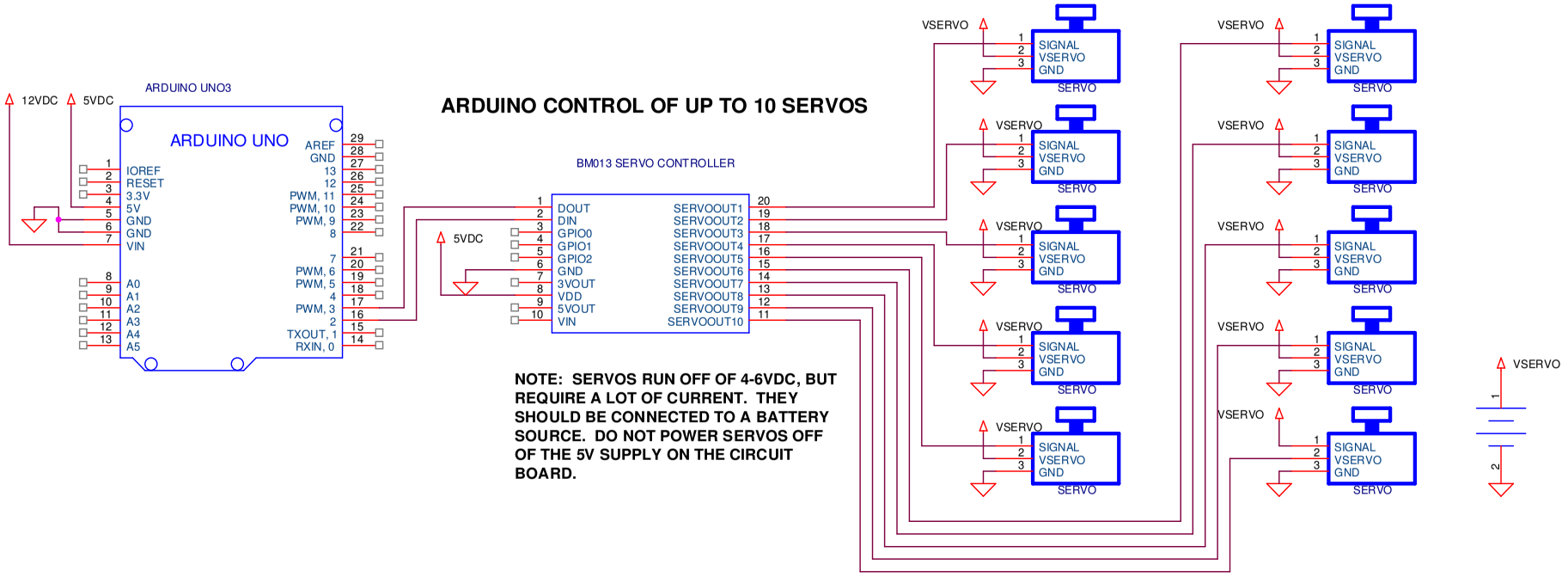
Indicator Register (index = 4): The Status register contains bit values that indicate important conditions in the module.

Indicator	Bit	Description
GPIO0_Input	0	GPIOx_Mode in Mode register must be set to 1. 1: Voltage at Pin 3 is 0.8VDD or higher. 0: Voltage at Pin 3 is 0.2VDD or lower.
GPIO1_Input	1	GPIOx_Mode in Mode register must be set to 1. 1: Voltage at Pin 4 is 0.8VDD or higher. 0: Voltage at Pin 4 is 0.2VDD or lower.
GPIO2_Input	2	GPIOx_Mode in Mode register must be set to 1. 1: Voltage at Pin 5 is 0.8VDD or higher. 0: Voltage at Pin 5 is 0.2VDD or lower.
	3	unused
	4	unused
	5	unused
	6	unused
	7	unused

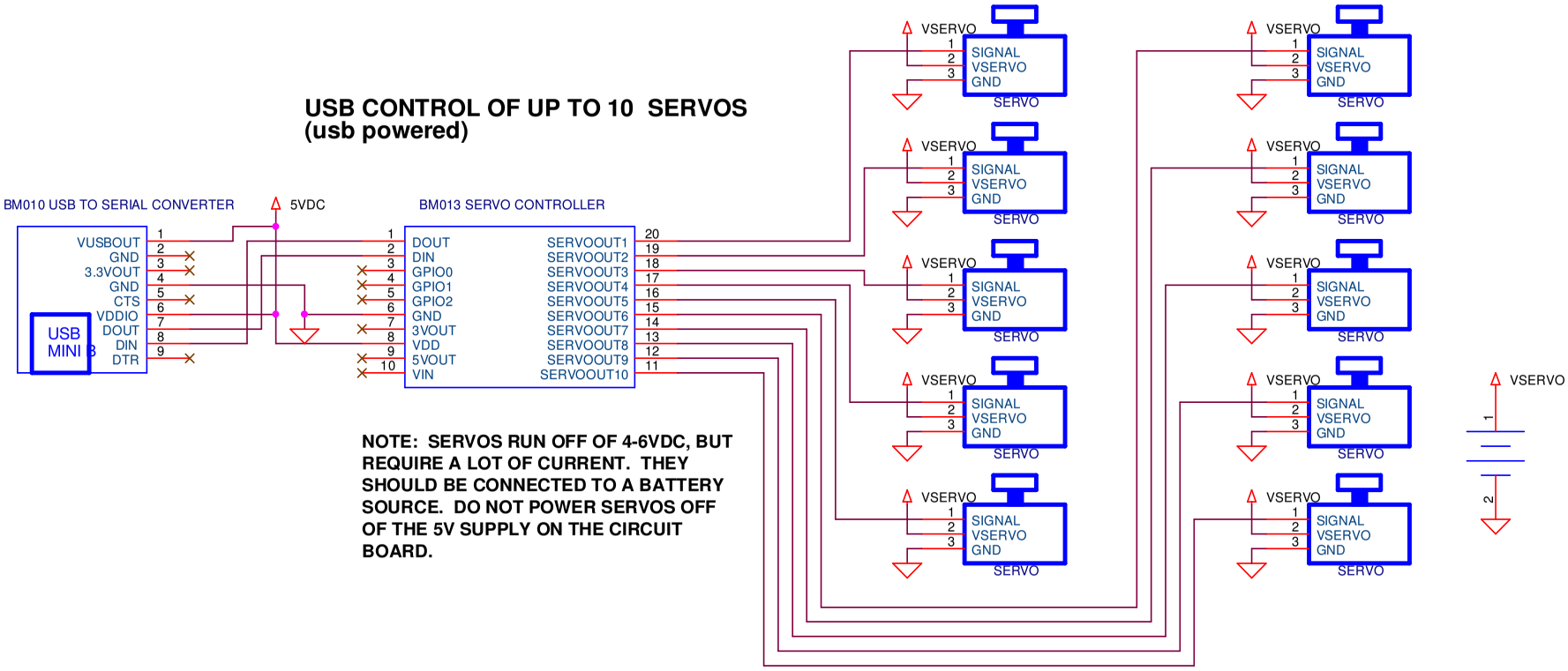
Schematics:


P1
DOUT
R1 270 Ohm
P2
DIN
R3 270 Ohm
P3
RC3
R6
270 Ohm
D1
Red
R8
1.0K
P4
RC6
R9
270 Ohm
D2
Red
R11
1.0K
P5
RC7
R12
270 Ohm
D3
Red
R14
1.0K
P6
GND
P7
3.3V
P8
VDD
P9
5V
P10
VIN
J2
WIRING POINT
VDD
R4
33K
SW1
NO momentary switch
MCLR
U1
PIC16F1829-I/ML
GNDTAB
RC7
RB7
RB6
RB5
RB4
RC2
RC1
RC0
RA2
RA1/PGC
RA4
RA5
VDD
VSS
RA0/PGD
RC6
RC3
RC4/TX
RC5/RX
MCLR
J1
PGRM/DEBUG
MCLR
VDD
GND
PGD
PGC
C1
0.1uF
R15
1.0K
D4
Red
U2
MIC5235-3.3
VIN VOUT
GND
EN ADJ/NC
C2
10uF
C3
0.1uF
R21
1.0K
D5
Red
C4
10uF
U3
MIC5235-5.0
C5
10uF
R2
270 Ohm
R5
270 Ohm
R7
270 Ohm
R10
270 Ohm
R13
270 Ohm
R16
270 Ohm
R17
270 Ohm
R18
270 Ohm
R19
270 Ohm
R20
270 Ohm
P20
RA2
P19
RA4
P18
RA5
P17
RC0
P16
RC1
P15
RC2
P14
RB4
P13
RB5
P12
RB6
P11
RB7
FID1
FID2
MH1
85 mil
MH2
85 mil

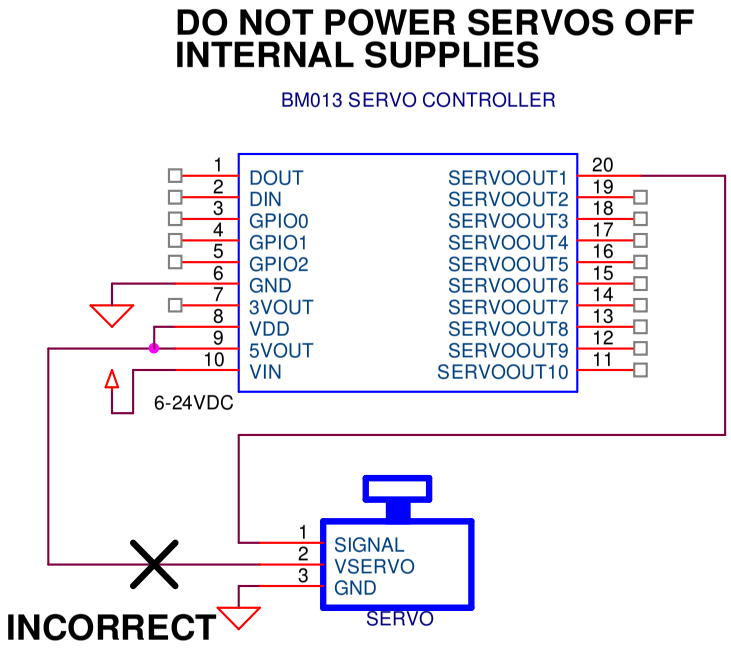
Application Schematics:

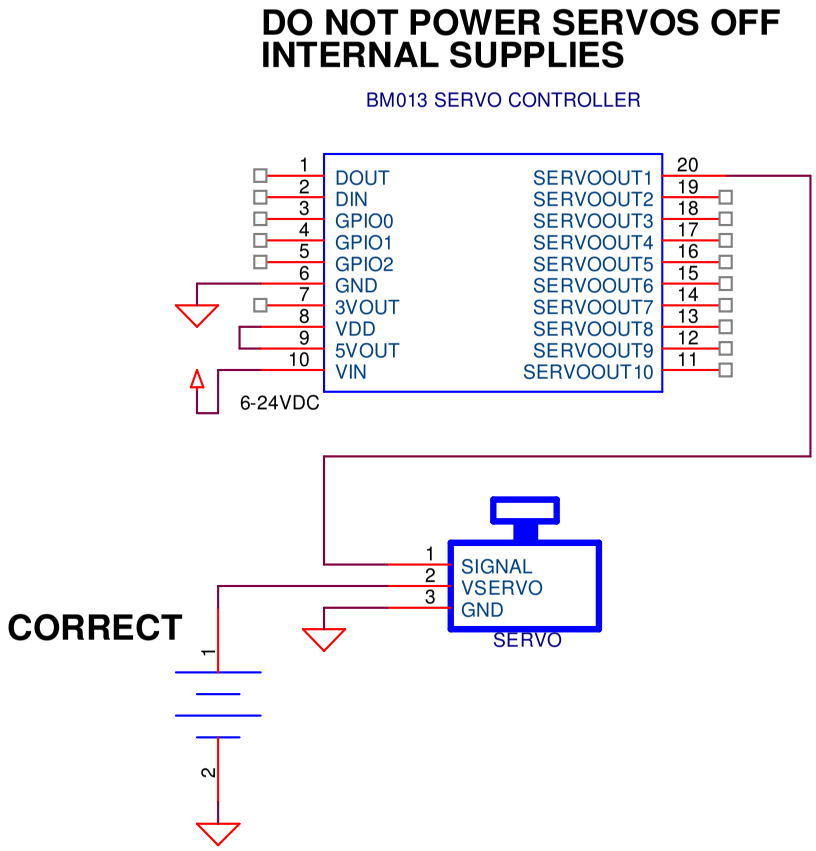


Application Schematics:



Application Schematics:





Application Schematics: