MAESTRO

Technical Manual

May 16, 2004

Introduction

Batteries are an essential component of many systems. They provide power for most portable systems, games, and entertainment devices of all kinds. Rechargeable batteries allow the consumer to re-energize their batteries after use and provide very good value to the consumer. In many applications loss of battery power is a nuisance causing no real harm, this is not the case in radio control (R/C) model aircraft. If the aircraft batteries "go dead" in flight it generally results in total destruction of the model and can represent a safety hazard. As a result, R/C modelers are very careful with their batteries. Most every experienced flier will test their battery status under load, just before flight to insure enough margin of safety remains for a flight. The MAESTRO's primary design goal is the monitoring of battery status, several additional utility functions are provided by the MAESTRO but battery condition is its intended role.

Many rechargeable battery chemistries result in discharge curves that are very "flat" with a steep knee in the curve when the energy stored in the battery is exhausted. An example of a 4 cell nickel cadmium pack is shown below and illustrates a typical discharge curve.

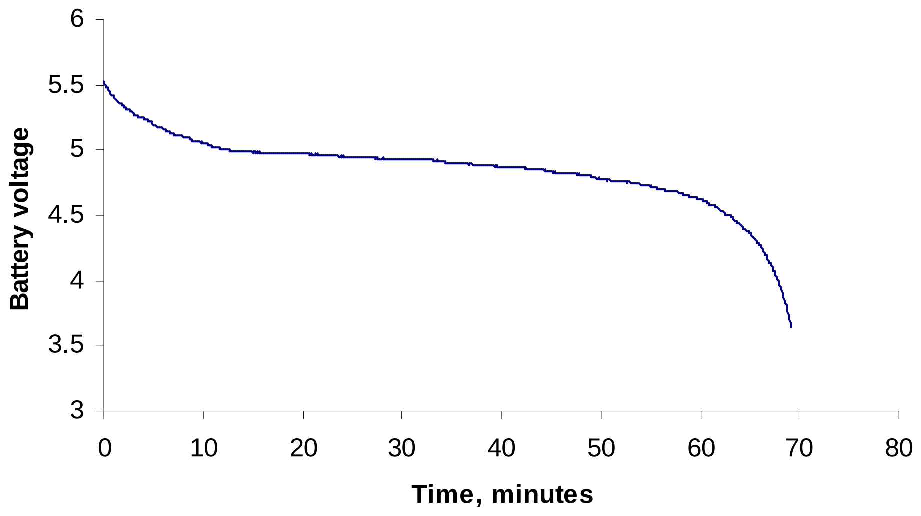


Figure 1, Discharge curve of a 4 cell NiCd pack, recorded with the MAESTRO system.

From this curve it is clear that the battery condition must be monitored carefully to catch this knee. A typical R/C modeler will own several different aircraft with rechargeable batteries used in the transmitter and the aircraft receiver. Different aircraft will use different battery capacities based on the size of the model. Different battery chemistries are also in use. Below is a list of the most popular batteries in use today:

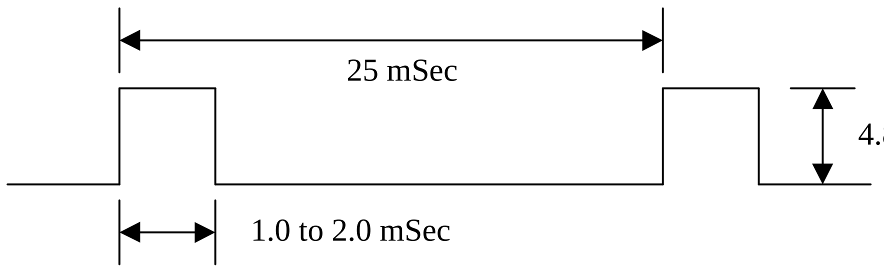
Chemistry	**Mnemonic**	**Cell voltage**
Nickel cadmium	NiCd	1.2V
Nickel metal hydride	NiMH	1.2V
Lithium Ion	Li-ion	3.7V
Lithium polymer	Li-poly	3.7V

Typical capacities of these cells vary from 100 milliamp hour to 2000 milliamp hour. Transmitters generally operate at 9.6 volts using 8 cell NiCd or NiMH packs. Airborne systems generally operate at 4.8 volts using 4 cell NiCd or NiMH packs. Often airborne systems operate using 5 cell NiCd or NiMH packs. The lithium battery technology is making significant inroads in the modeling community due to there incredible power density. Lithium packs are constructed using two cells in series resulting in a 7.4 volt pack. This requires a voltage regulator to operate the airborne system in specification.

This heterogeneity in battery chemistries, capacities, and configuration begs for a programmable testing system that can be configured to the users needs. MAESTRO fills all of these needs.

The battery monitoring feature is the primary purpose of the MAESTRO but several additional features have been added. These features allow you to control the position of a servo, and monitor pulse widths that are sent to a servo to control its position. These operations are performed using the timer function of the Atmel microcontroller and its capture and compare registers to generate or monitor the pulse widths.

All radio systems consist of a transmitter the pilot holds to fly the airplane and a flight pack mounted in the model. Servos are used to move the control surfaces and they receive position information from the receiver. The servos connect to the receiver with a three-wire interface, ground, power, and the position information. The servos typically run from a 4 cell NiCd battery pack, nominally 4.8 volts. The position information is a pulse stream with the pulse width determining the servo position. 1.5 mSec is the center position, 1.0 mS to 2.0 mS define the full range of motion of the servo. The position pulse is sent to the servo every 25 mS, or at a 40 Hz rate. A timing diagram for the servo signal is shown below:



MAESTRO contains several functions to generate servo control signals and monitor the pulse width information sent to the servos. These functions are useful for setting up a new aircraft and performing system maintenance. More details on these functions can be found in the operations manual.

Design overview

At the heart of the MAESTRO is an Atmel mega8 microcontroller. The schematic is shown in figure 1. The circuit is very simple consisting of the following main functional blocks:

Voltage regulation, IC2

A simple linear voltage regulator is used to provide a stable 4.0 volt supply to power the microcontroller and LCD display. This power is taken from the battery being tested; this requires that the battery provide a minimum of 4.0 volts. This is a low drop out regulator so you can get functionality down to 4.0 volts. The system will operate and hold calibration to approximate 3.0 volts, the display will be hard to see at this reduced voltage but the unit will still function.

Microcontroller, IC1

The Atmel mega8 was selected for this design. This microcontroller contains the peripherals needed for the MAESTRO. The main features needed include:

- Analog to digitial converter, ADC. The ADC is used to measure the input battery voltage. R3 and R4 form a voltage divider to reduce the battery voltage to a measurable range. The 10 bit ADC of the mega8 provides the resolution needed for this application.
- Internal voltage reference. The mega8 internal voltage reference enables accurate detection of the battery voltage without the need for an external reference.
- Timer functions allow the generation of pulses used to drive R/C servos and monitoring of pulse widths use to monitor servo control signals. These timers are also used to generate a real time interrupt that control all the time specific functions.
- The uart is used to provide an interface to the PC host application. This host application allows the user to define operating parameters and communicate with the MAESTRO. The PC host application will even allow erasing and reprogramming the flash memory of the MAESTRO. The three wire interface of the MAESTRO can be connected to a simple interface to allow connection to a conventional RS232 comm port of your PC. This interface on the mega8 side is very simple, the receive and transmit lines of the uart are tied together. When the MAESTRO needs to receive, the transmitter output is disabled.

LCD display

The MAESTRO uses a 4 line 12 character per line LCD display. This is a surplus display that I found on line at http://earthlcd.com/. There is a huge quantity of these displays available and the cost is very low when purchased in quantity, as a result most of my projects use this display! The interface requires 8 data lines and 4 control lines. J1 provides the interface to the LCD. This connector has 2 mm pin spacing and the flexible connector plugs directly into this connector.

Load, T1, R7

The programmable load is implemented using a 5 ohm power resistor and a NPN transistor. The transistor is controlled by the mega8 to vary the duty cycle of the load. At 100 percent duty cycle the load is 5 ohms, at 50 percent duty cycle the load is 10 ohms. A timer channel of the mega8 is used to enable this pulse width modulation of the load resistor. The load is rated at only 3 watts and is not intended for long term loading of the battery pack. The idea is to apply the load during the few seconds it takes to assess battery state. The MAESTRO will not allow you to leave the load on for extended periods of time.

User controls, SW1, D1

The user control consists of one button, that's it! This button connects to an IO pin of the microcontroller. The IO pin is pulled up to 4 volts using the internal pull-up feature of the mega8. The momentary push button shorts this pin to ground. Multiple functions are used with this button, different function are selected by the length of time the button is held and by pressing the button multiple times to signal special actions. A single LED is also driven by the mega8, this LED provides a visual warning when the load resistor is enabled.

The hardware design is very simple; the remaining circuitry includes the oscillator, decoupling capacitors, and the in-circuit-programming connector, SV1. Two interface connectors are used, one male and one female. The two connectors are factually identical.

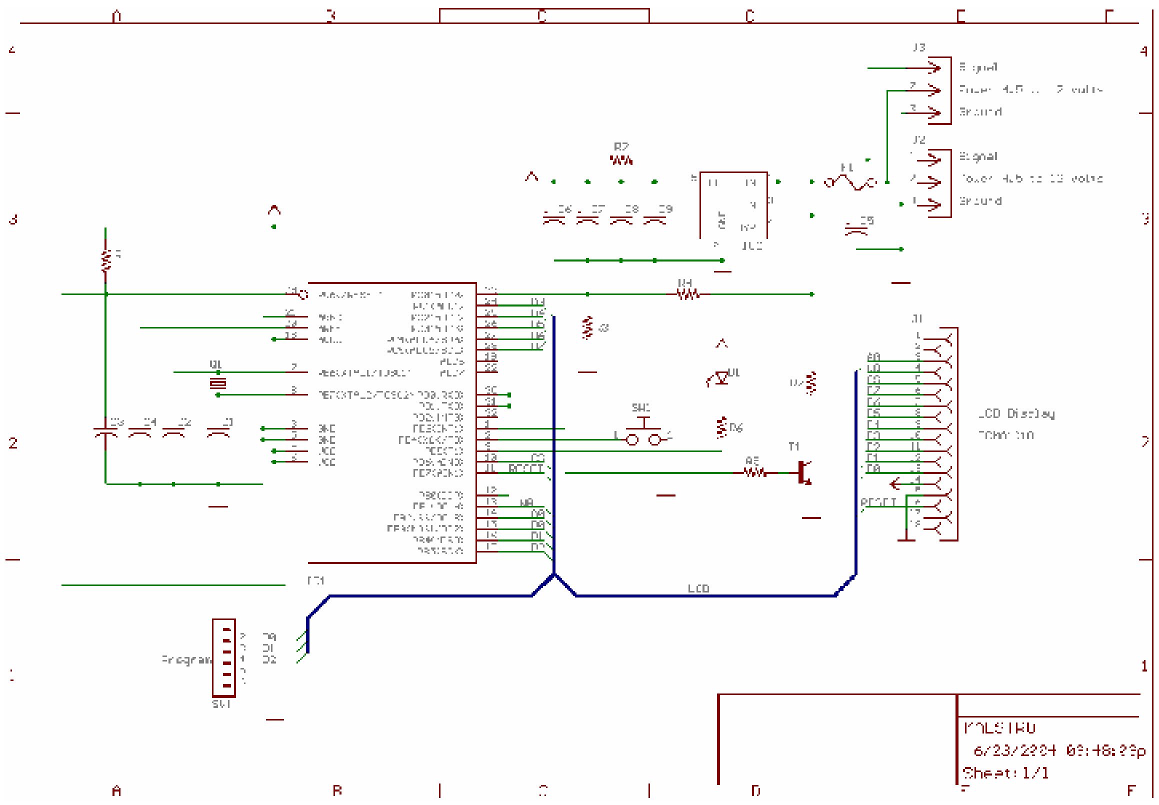


Figure 2, Schematic diagram of MAESTRO.

The MAESTRO is housed in a small plastic case, 2.4" x 1.85". The electronics are contained on a single printed circuit board (PCB). The PCB layout is shown in figure 2. The 44 pin TQFP package was selected for the mega8, surface mount components were

used for most components on this design to reduce the size. The LED and pushbutton switch are both mounted on the PCB.
The display connector, J1, is positioned to allow the LCD to plug directly into the PCB, this make the assembly of the MAESTRO very simple. No internal cabling is required to fabricate the MAESTRO.

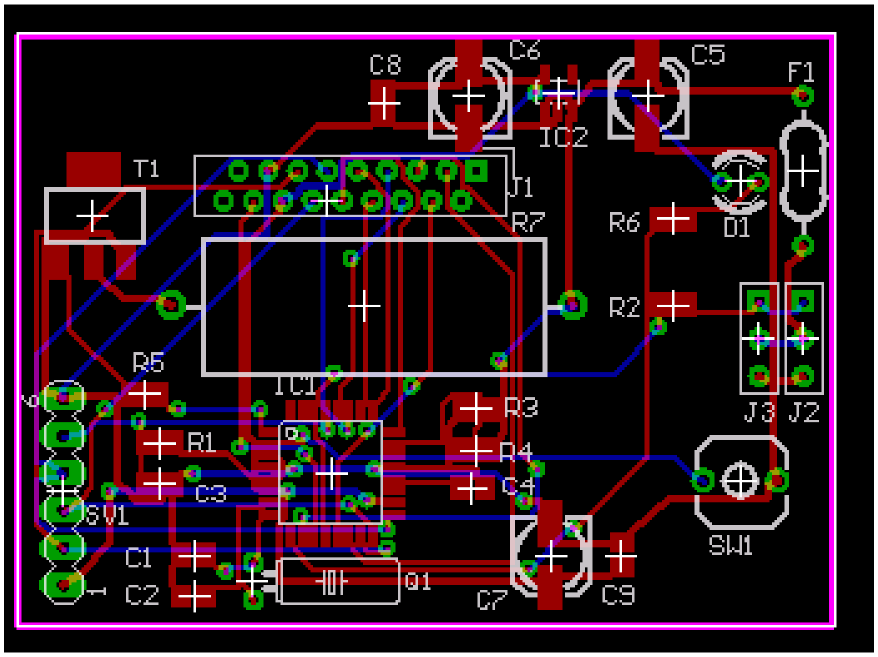


Figure 3, MAESTRO circuit board layout

The full parts list for the MAESTRO is shown in table 1. The parts selected for this design are readily available and suggested suppliers are shown on the parts list.

Item	Quanity	ID	Part number	Description	Supplier	Cost each	Total cost
1	2	C1, C2	140-CC501N220J	22pF 0805 NPO	Mouser	0.08	0.17
2	4	C3, C4, C8, C9	140-CC501B104K	0.1uF 0805	Mouser	0.14	0.56
3	3	C5, C6, C7	PCE3180CT-ND	33uF 16V electrolytic	Digikey	0.26	0.78
4	1	D1	67-1064-ND	LED	Digikey	0.12	0.12
5	1	F1	F828-ND	PICO II 2 amp fuse	Digikey	0.73	0.73
6	1	IC1	ATMEGA8L-8AC-ND	8-Bit microcontroller	Digikey	5.27	5.27
7	1	IC2	LP2981AIM5-4.0CT-ND	4.0 Volt 150 mAmp regulator	Digikey	1.24	1.24
8	1	J1	HFF18T-ND	1mm Flex connector	Digikey	0.95	0.95
9	2	J2, J3		Cable connections			0.00
10	1	Q1	SE3414-ND	8MHz cylinder crystal	Digikey	1.17	1.17
11	1	R1	260-10K	10K 0805 resistor	Mouser	0.08	0.08
12	2	R2, R5	260-100	100 0805 resistor	Mouser	0.08	0.16
13	1	R3	260-1K	1K 0805 resistor	Mouser	0.08	0.08
14	1	R4	260-4.7K	4.7K 0805 resistor	Mouser	0.08	0.08
15	1	R6	260-470	470 0805 resistor	Mouser	0.08	0.08
16	1	R7	45F5R0-ND	5.0 ohm 5 watt resistor	Digikey	1.37	1.37
17	1	SV1		Programming connection			0.00
18	1	SW1	P8081SCT-ND	Light touch switch, momentary	Digikey	0.38	0.38
19	1	T1	FZT649CT-ND	NPN power transistor	Digikey	1.38	1.38
20	1		SCR8A-ND	1.85x2.40 C series SERPAC	Digikey	3.24	3.24
21	1		Maestro V1.0	PCB rev 1.0		2.00	2.00
22	1		ECMA1010	4x12 LCD display	Earth Tech	3.00	3.00
23	1		215FJ24	24" Futaba J Extension Connector	Cermark	4.00	4.00
24	1		260-220K	220K 0805 resistor used on display	Mouser	0.08	0.08
25	1		P11337CT-ND	10uF 6.3V solder on top of R3	Digikey	0.77	0.77
					Grand total		27.69

Table 1, MAESTRO parts list

Software design

Three software development efforts were required to develop MAESTRO. The first component was the development of the bootloader program to allow the MAESTRO application to be loaded into flash through the serial interface. The second application was the development of the MAESTRO application. The third development effort was the PC host application used to communicate with the MAESTRO.

Bootloader

The bootloader was developed using Atmel AVR Studio 4 all software was written in assembly language.

The bootloader application was adapted from the Atmel application note (AVR109 Self-programming) available from their web site. This bootloader was designed to reside in the boot section of flash memory and communicate through the serial port to the AVRprog application. This bootloader program would not work in my application because of the hardware design issues plus I needed to add additional options to allow the bootloader to detect the presents of the MAESTRO application and vector to it on power up.

The MAESTRO contains a three wire interface, power, ground and one signal lead. This signal lead has many functions including serial IO. The serial IO is performed by tying the serial send and receive leads together and connecting these leads to the signal line. This means that the transmitter needs to be disabled to allow the receiver to input characters. The serial IO line in this application is bi-directional. The bootloader application was changed to support this mode.

When the MAESTRO is powered up, the bootloader application starts. The first thing the bootloader does in look at the last two bytes, or the last word, in the application area. If a signature work is found then the boot loader assumes the application has been loaded and the bootloader vectors to the application.

The application can vector to two different locations in the bootloader to support reprogramming the application. The erase entry point will erase the application area and then start the bootloader program. The program entry point will start the bootloader but not erase the application area. The MAESTRO has a command to allow you to jump to the program entry point.

The AVRprog application will not work to communicate to the bootloader because of the bi-directional serial IO feature. The MAESTRO PC host application contains a programming function that will allow the user to erase and reprogram the MAESTRO. Additionally a simple serial adapter is required to convert the RS-232C levels into TTL bi-directional interface. This interface uses a MAX232 and one PNP transistor.

Ad additional commend was added to the bootloader to allow the PC application to start the MASTRO application. This is the go command, “G”.

MAESTRO

The MAESTRO application was written using the Atmel AVR Studio 4 development system, all software was written in assembly language. Several Atmel application notes were used to support 16 bit arithmetic.

MAESTRO contains a main polling loop plus interrupt service routines to support its functions. The roll of each of these components of the software design are describer below:

Timer 1

Timer 1 is the used to provide a 25 milli-second real-time interrupt for the MAESTRO. This real-time interrupt is used for all timing functions. This timer is also used to measure pulse widths using the capture mode and to generate servo driving pulses using the compare register to generate an interrupt and toggling IO lines. The timer 1 service routine sets a flag when an interrupt occurs; this happens every 25 milli-seconds and is called the tick rate.

Timer 2

Timer 2 is used to generate the pulse width modulation signal to drive the load resistor. The MAESTRO has a programmable battery load. Oregon Aerobatic Challenge The way this works is a fixed load resistor, 5 ohms by default, is enabled and disabled with a transistor. If the desired load is 10 ohms then the transistor is pulse width modulated at 50 percent duty cycle. The transistor is controlled through an interrupt service routine.

Serial receiver

The serial input data is processed through an interrupt service routine. All received characters are stored in a buffer. After an end of line character is received a flag is set to inform the main processing loop that a command needs to be processed.

Polling loop

The main polling loop of the MAESTRO performs all major functions. The serial input data are processed at the 25 milli-second tick rate and all other functions are performed at the frame rate that has been set at 250 milli-seconds. A set of flow diagrams have been included that define the operating flow of the MAESTRO application.

The MAESTRO uses a set of utility functions that are very useful in other applications. The following section describes a few of these utility functions:

LCD interface

I'm using a surplus LCD display designed for a cell phone application. This display has 4 lines of text with 12 characters per line. It also contains a battery level icon that is perfect for this application plus a number of other icons that could be used in a number of different applications. These are very economical displays, when purchased in 800 quantities the cost is only 50 cents per display. As a result I use this display is most projects. The display can be modified to support a simple serial interface when limited IO pins are available. I am using its standard 8 bit parallel IO interface.

The software drivers I have developed allow the standard set of necessary functions including; initialization, character output, string output and control functions of clearing the display etc.

The formatted IO function can send results to the serial port or to the LCD display; this is accomplished using a flag to indicate the output device.

Formatted IO

A table driven formatted IO system has been developed to support all of the serial IO commands used in the MAESTRO. This design has been used in several projects and makes it very simple to add addition IO commands. The table CTBL contains all of the IO commands supported through the serial interface. Below is an example table entery:

```
        .db     "DMD ", 1,0,low(Mode),high(Mode)
```

The table consists of the following elements:

- The first four bytes are the command label. Received serial commands are parsed and the table is searched for a command match. The search is case sensitive.
- The next byte defines the IO type. The following types of IO operations are supported

    1 = Byte input/output
    2 = Word input/output
    3 = Jump to function routine
    4 = Byte output
    5 = Word output
    6 = Indexed byte IO

7 = Indexed word IO
8 = Selection table
9 = Display table value
10 = Send string, string from flash
11 = Float input/output (24 bit floats)
12 = String input/output, string in ram
13 = Seconds input/output
14 = Float input/output (16 bit floats)

- The next byte contains flag bits. The flags are not used in this application.
- The last two bytes form a word pointer to the variable, for example; if this is a word IO function then this word would contain the address of the word you wish to use for the input and output function.

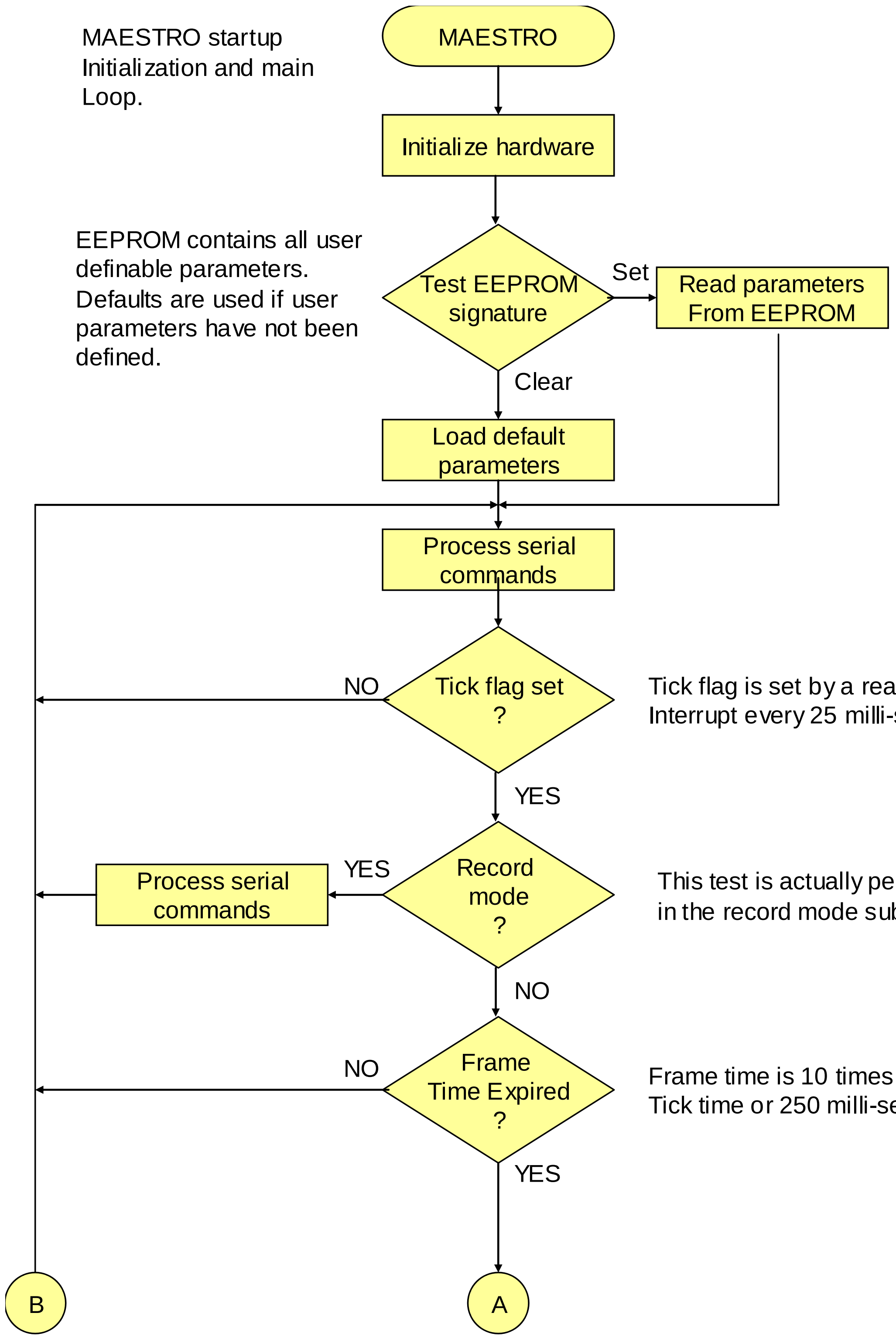
Floating point arithmetic

The MAESTRO uses a simplified floating point arithmetic system. All floating point numbers are really integers multiplied by 100 with an implied decimal point. This provides the necessary precision and was very simple to implement.

The following pages document the MAESTRO software design through a series of flow diagrams. This set of diagrams does not document every routine in the system, just the main functions and logic flow.

MAESTRO startup
Initialization and main
Loop.

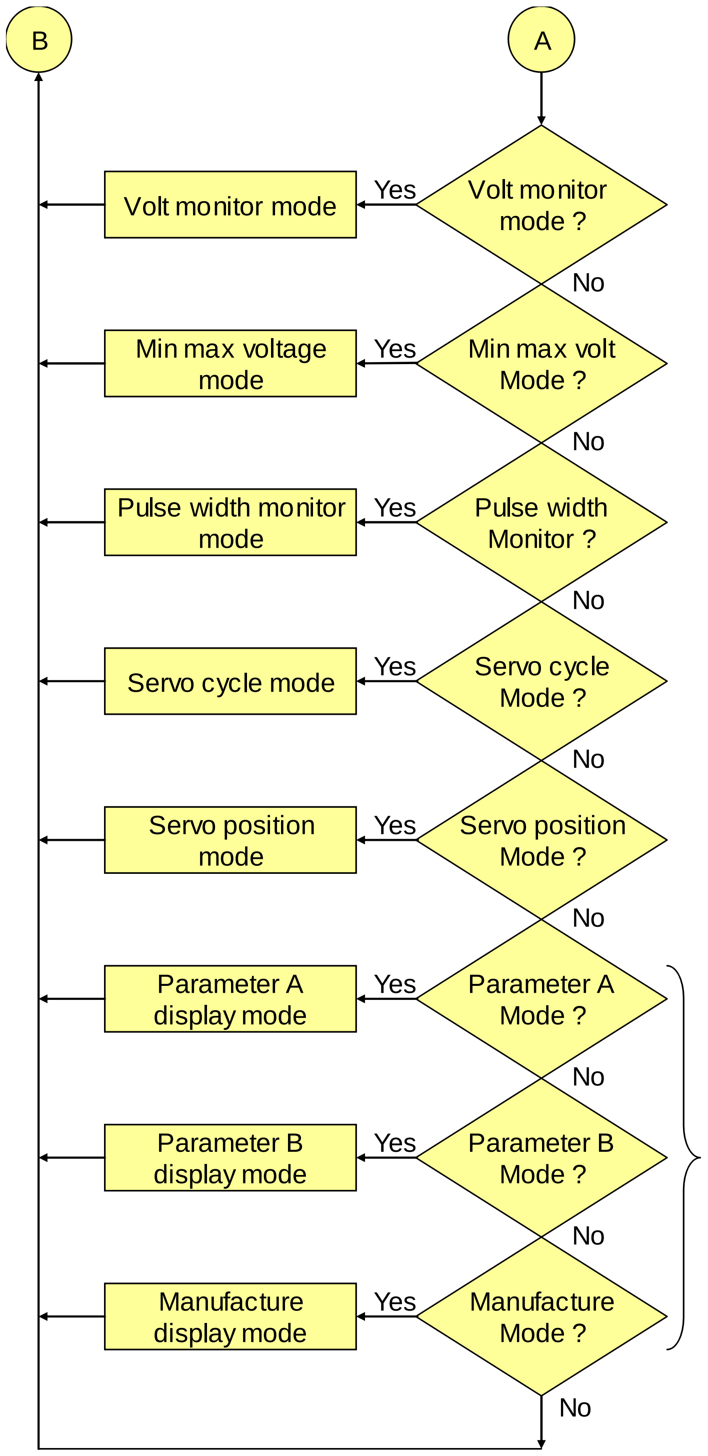
EEPROM contains all user
definable parameters.
Defaults are used if user
parameters have not been
defined.



Tick flag is set by a real-time
Interrupt every 25 milli-seconds.

This test is actually performed
in the record mode subroutine.

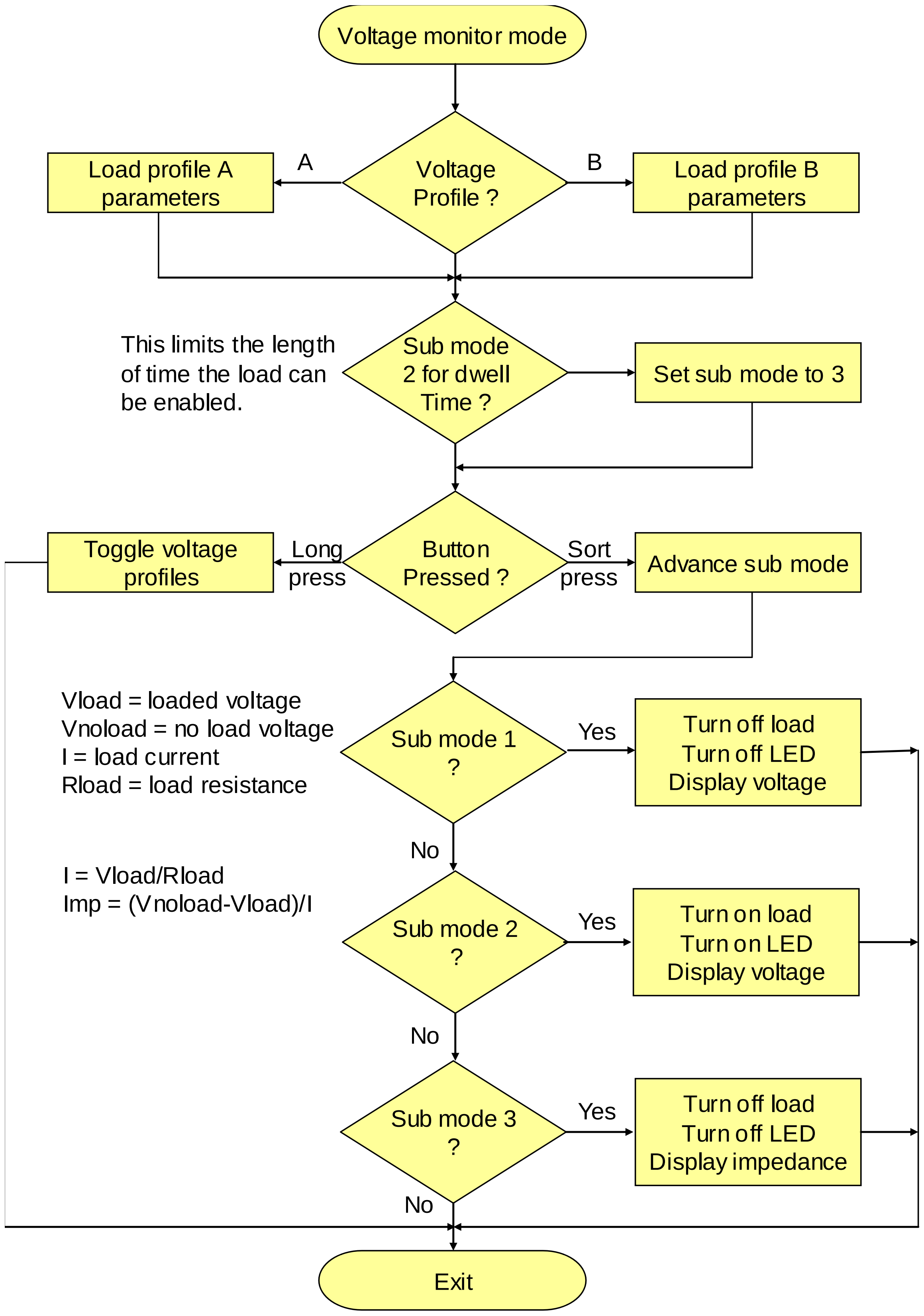
Frame time is 10 times the
Tick time or 250 milli-seconds.



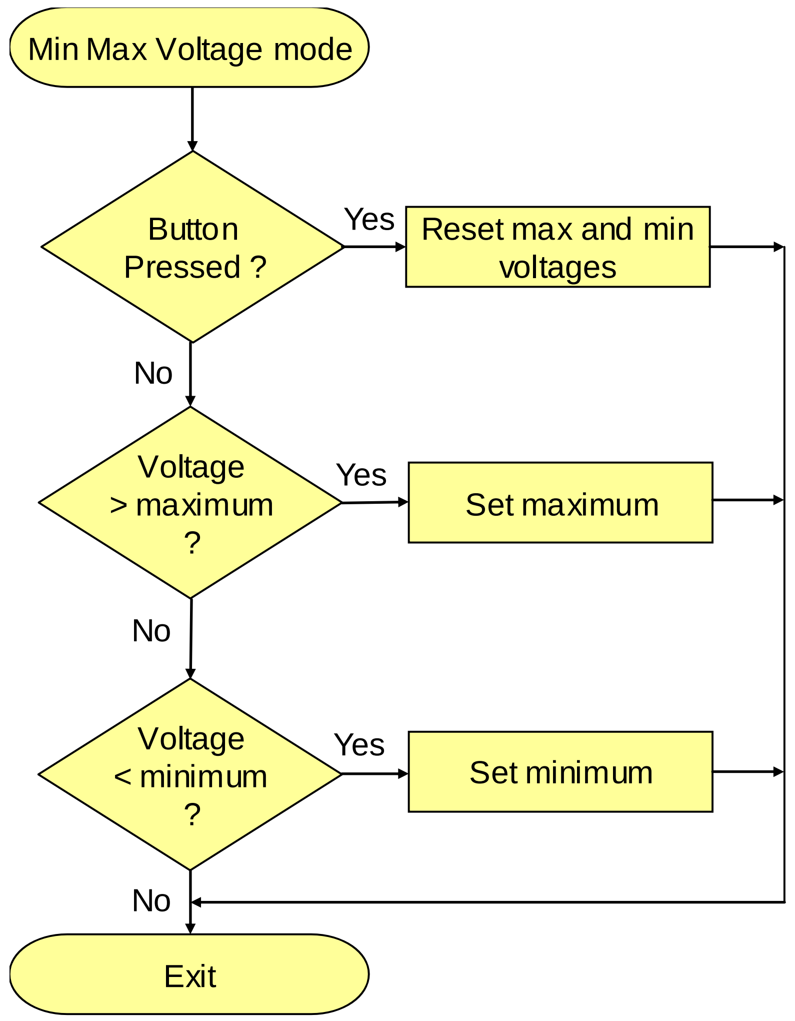
The mode flag is read to determine the operating mode. This test is actually performed in the specific mode subroutine, its shown here to reduce complexity in the mode flow charts.

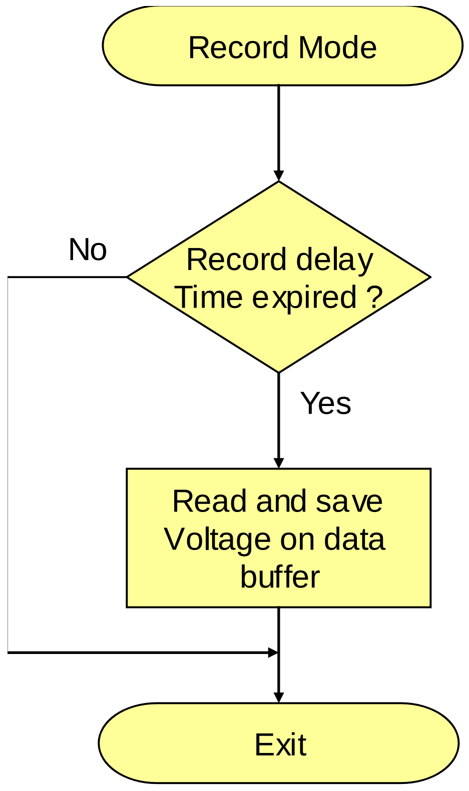
The pulse width monitor mode is very simple and the flow logic will not be shown. This mode uses the capture mode to read the pulse width and displays the value.

These are display modes only. Flow logic will not be shown for these routines because all they do is display parameters on the LCD screen for reference.



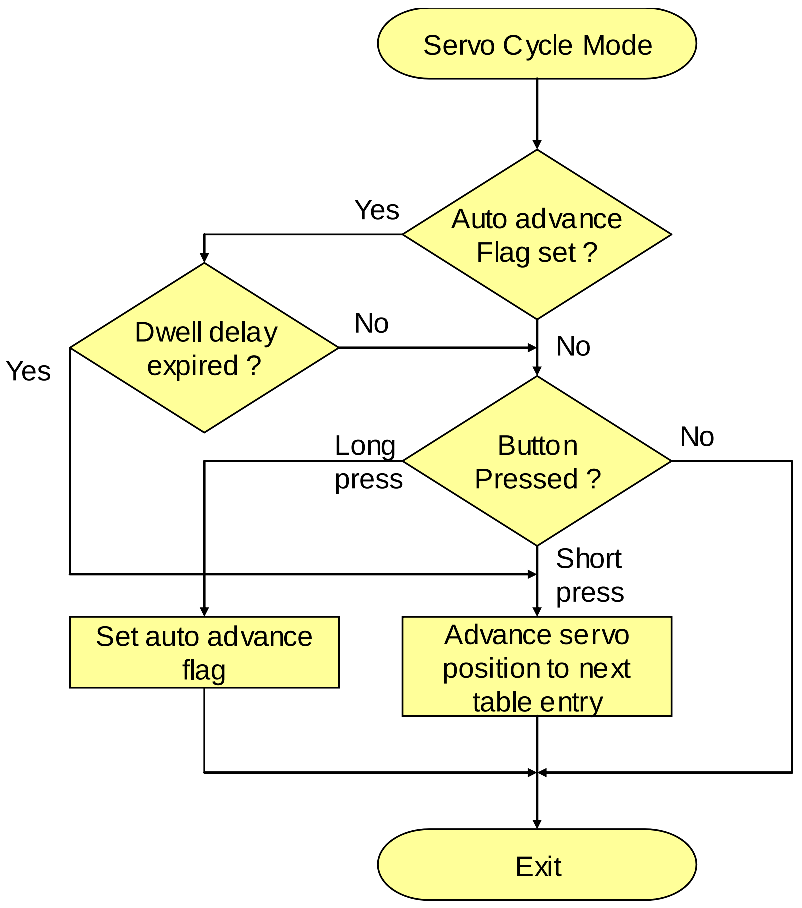
Voltage monitor mode
Voltage Profile ?
A
B
Load profile A parameters
Load profile B parameters
This limits the length of time the load can be enabled.
Sub mode 2 for dwell Time ?
Set sub mode to 3
Toggle voltage profiles
Long press
Button Pressed ?
Sort press
Advance sub mode
Vload = loaded voltage
Vnoload = no load voltage
I = load current
Rload = load resistance
Sub mode 1 ?
Yes
Turn off load
Turn off LED
Display voltage
No
I = Vload/Rload
Imp = (Vnoload-Vload)/I
Sub mode 2 ?
Yes
Turn on load
Turn on LED
Display voltage
No
Sub mode 3 ?
Yes
Turn off load
Turn off LED
Display impedance
No
Exit



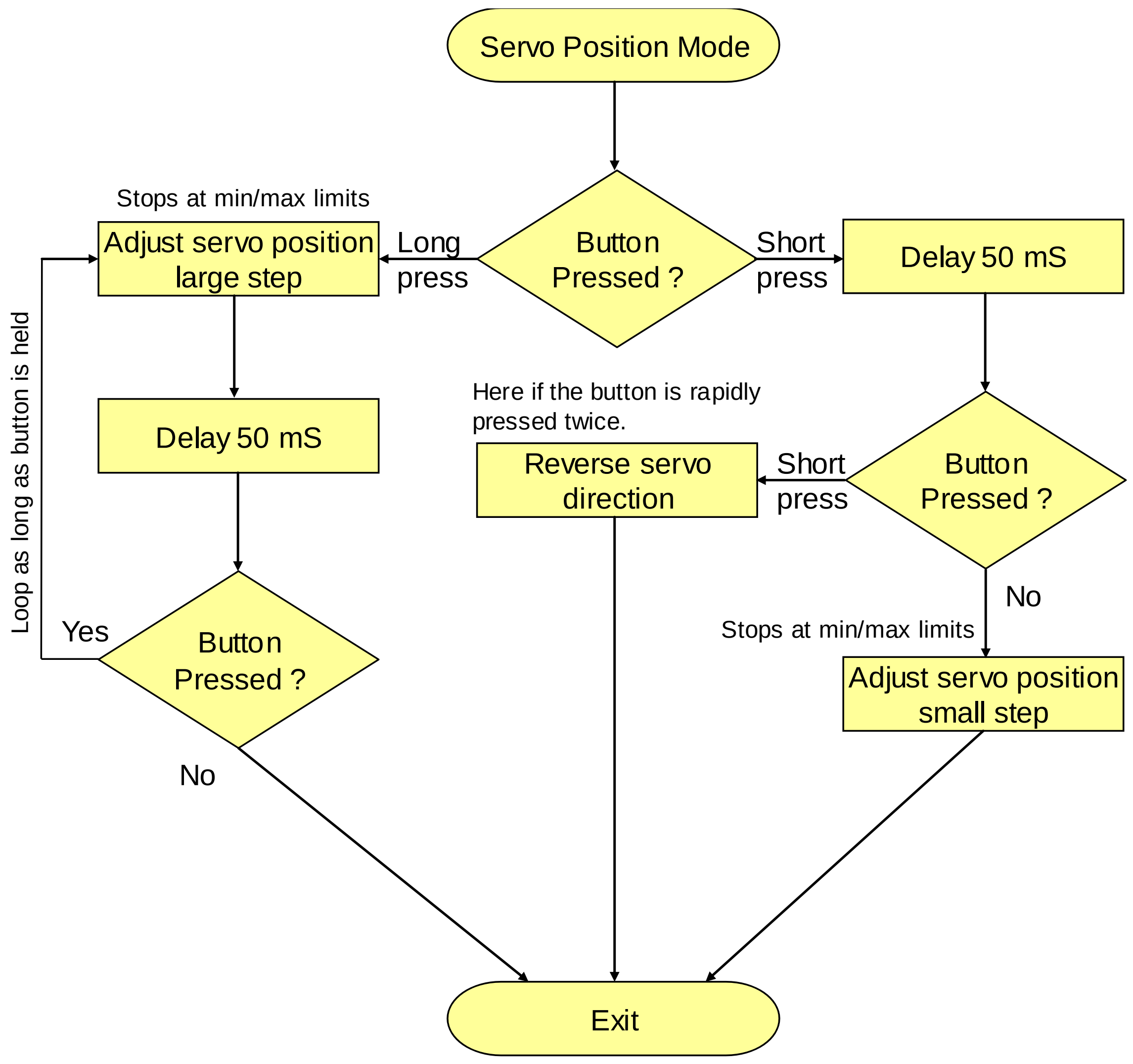

Record Mode
No
Record delay Time expired ?
Yes
Read and save Voltage on data buffer
Exit

The record delay time is programmable with a resolution of 25 milli seconds.

The data buffer size is 300 words so 300 voltage values can be saved in the buffer. The data storage will stop and the LCD display will show FULL when the buffer reaches its end.



Servo cycle mode has a table of ten servo positions. The advance function will move through the table and reset to the start after the tenth position or when a -1 flag is seen in the table.

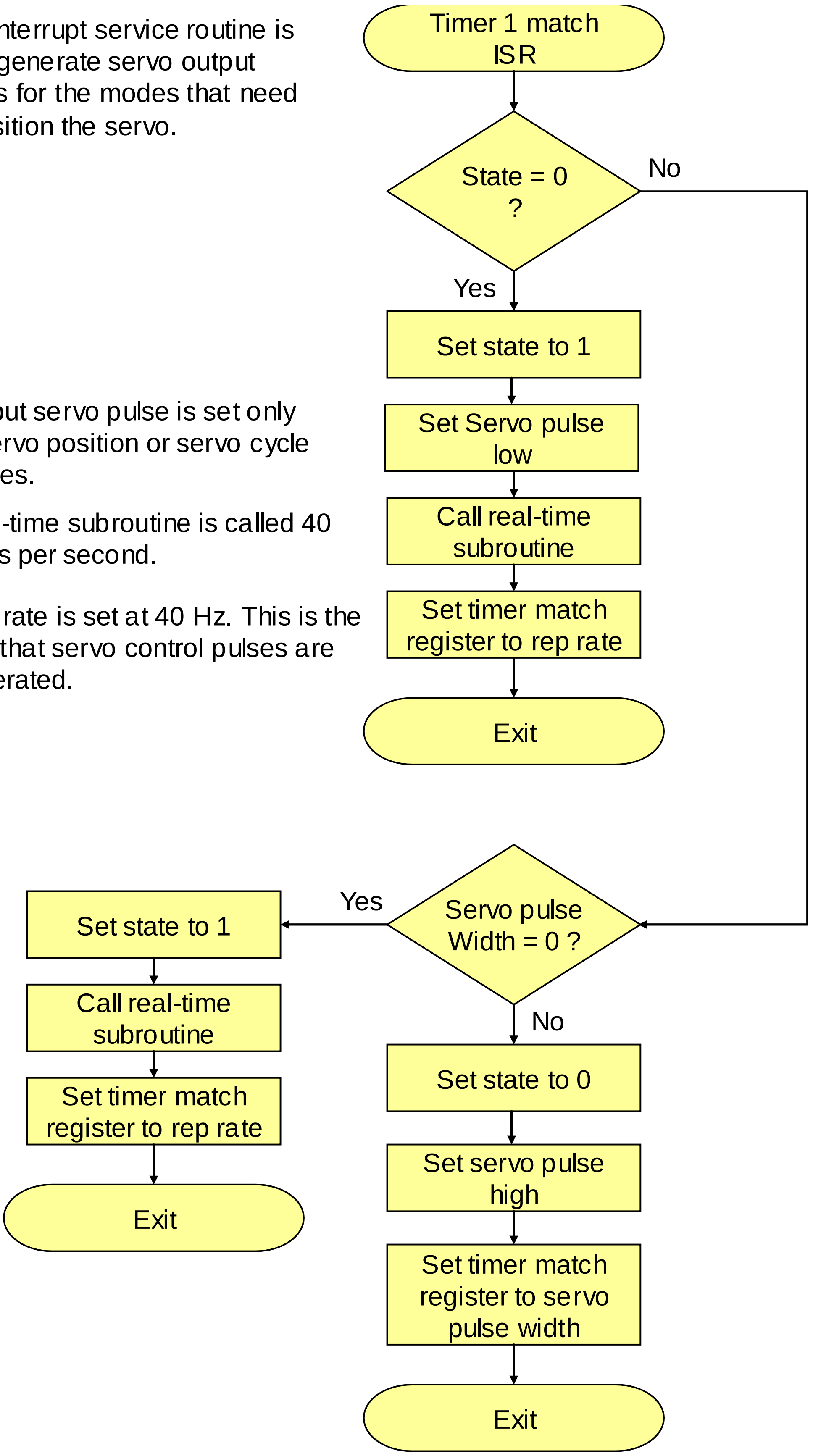


This interrupt service routine is used generate servo output pulses for the modes that need to position the servo.

Output servo pulse is set only in servo position or servo cycle modes.

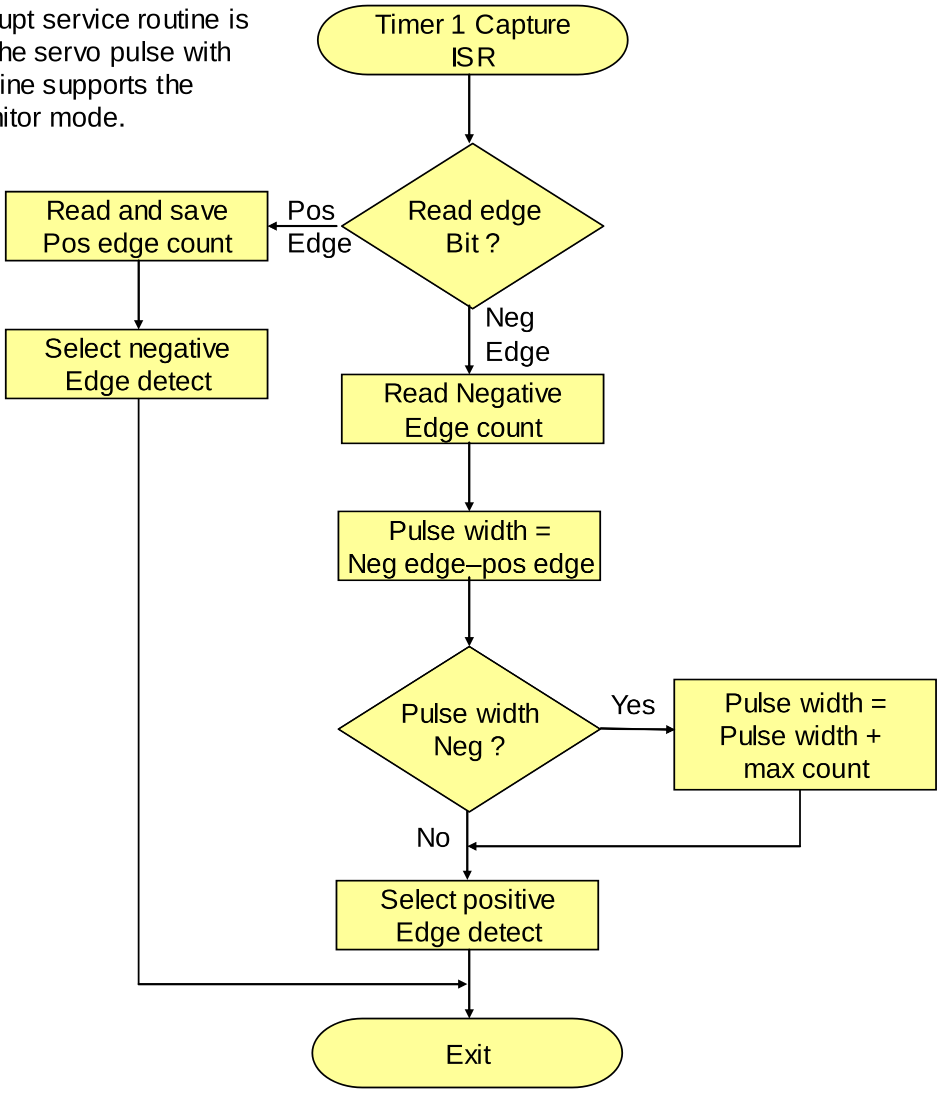
Real-time subroutine is called 40 times per second.

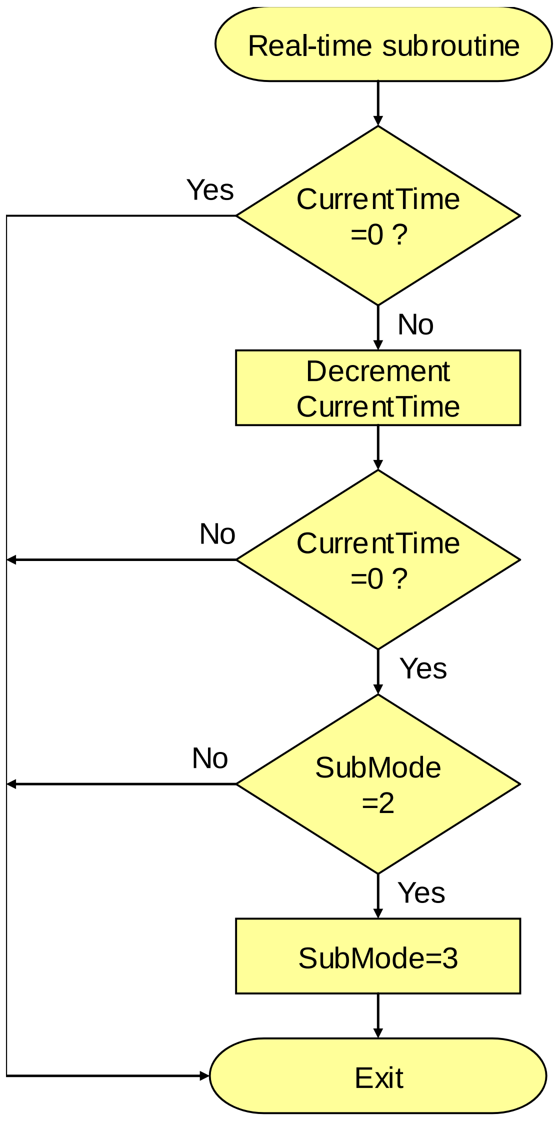
Rep rate is set at 40 Hz. This is the rate that servo control pulses are generated.



Timer 1 match ISR
State = 0 ?
No
Yes
Set state to 1
Set Servo pulse low
Call real-time subroutine
Set timer match register to rep rate
Exit
Servo pulse Width = 0 ?
Yes
Set state to 1
Call real-time subroutine
Set timer match register to rep rate
Exit
No
Set state to 0
Set servo pulse high
Set timer match register to servo pulse width
Exit

This timer interrupt service routine is used to record the servo pulse with signal. This routine supports the pulse width monitor mode.


Timer 1 Capture ISR
Read edge Bit ?
Pos Edge
Read and save Pos edge count
Select negative Edge detect
Neg Edge
Read Negative Edge count
Pulse width = Neg edge–pos edge
Pulse width Neg ?
Yes
Pulse width = Pulse width + max count
No
Select positive Edge detect
Exit



This subroutine is used to control all of the timed function. The variable CurrentTime is defined in the main loop and this function will decrement the variable till zero. This routine is called at the tick rate, 40Hz. This allows function to have well defined delay and dwell times.

The SubMode is part of the voltage monitoring function. SubMode 2 means the load is on. When the CurrentTime limit is reached the load is automatically turned off by setting the SubMode to 3. This is a safety function to keep the total energy dissipation low.

PC host application

The PC host application is written in Microsoft Visual Basic version 6.0. This is a very simple application using the Microsoft comm object to communicate with the MAESTRO through the serial interface device. This application is described in the user's manual and it uses the serial commands also described in the MAESTRO user's manual. The programming option of this application is of general interest and can be adapted to many different applications.

Contact information

Gordon Anderson
POB 335
1104 Christopher Lane
Benton City, WA 99320
GAA@owt.com
www.mstar2k.com
(509) 588-5410