EE3102

9 March 2012

University of Minnesota

Design Report

Project Members:
Patrick Barrett
Mark Pennebecker
Kayiita Johnson
Prashant Dhakal

Submitted to:
Professor Higman

Abstract

The general purpose of this project is the design and construction of a music tuner. Specifically, this tuner will be checking whether or not an input pitch matches 440Hz (A). The input will be read from either a microphone or a coax cable fed from an oscilloscope. The purpose of a tuner is not only to test on whether or not a pitch matches a note, but also to see if the input pitch is sharp or flat (whether the pitch is above or below the given pitch). Our tuner will indicate this information through a series of LEDs. We plan to use a microcontroller to analyze the input and control the output LEDs.

1 Introduction

The methodology for finding pitch in our tuner is largely based upon the theory behind musical tones and how those pitches are measured.

The pitch we are measuring is called A440, which is a standard tuning pitch for musical instruments. The 440 denotes the frequency in Hz of the pitch, which is called A by convention. An octave is defined as a ratio of 2:1 between 2 pitch frequencies. In this case, the pitch A which is an octave above has a frequency of 880 Hz, while the pitch A below the standard A has a frequency of 220Hz. Each octave is split into 12 'semitones', which are considered distinct pitches within the octave. Since we are trying to determine if the input is in tune (matching the 440Hz frequency) we require further precision past the semitone measurement. Thus, we need to measure the accuracy of the input pitch using the unit known as the cent. A cent is a further division of the interval between 2 semitones. The interval between 2 semitones is divided into 100 cents. This means that each octave is divided into a total of 1200 cents and that the cent represents a different range of frequency depending on the octave.

We were tasked with measuring the difference between the input pitch and A440 in terms of 5 cent intervals. This specification of the interval of measurement was given by the project requirements, but it does make sense given that there is little audible difference between adjacent cent frequencies.

2 Overview

The basic flow of the signal is shown in Figure 1. After the input is received from the function generator (microphone input will need a few more steps), the signal will pass through a bandpass filter to help remove any noise from the signal. After being filtered, the signal will pass through a preamp, to boost the signal so that it can be further processed by the later stages. The signal is then sent to a square wave converter, which will take the analog signal and convert it into a series of square pulses to be handled by the microcontroller. The microcontroller will take the signal generated from this stage and measure the frequency of the square pulses over a certain period of time. Using this information, the microcontroller will choose light up an LED array to display how close the signal is to the desired pitch.

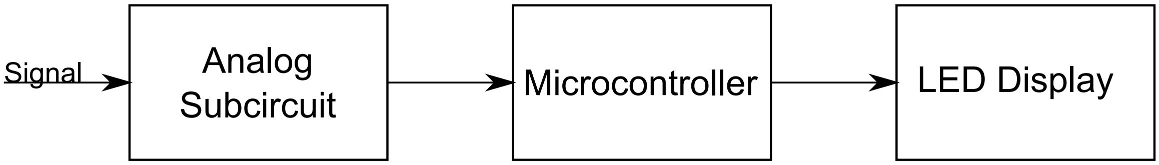


Figure 1: Block Overview

3 Analog Subcircuit

The analog portion of the circuit is a single IC consisting of four operational amplifiers, three of which are used. See Figure 2 for overview of the usages, where each stage uses a single opamp. The Pre-Amp is simply a voltage follower with a blocking capacitor and a voltage divider on the input to reference the incoming signal to 1/2 V_{CC}. The Filter is a simple inverting amplifier configuration with filtering caps which creates a band pass filter. This was designed to have an $f_{cH} = 880$Hz and an $f_{cL} = 220$Hz. Those frequencies being an Octive above and below the center frequency respectively. The Square Wave Generator is simply configured as a comparator refrenced to 1/2 V_{CC}. The output of the comparator is then fed to the microcontroller to be processed further and interpreted.

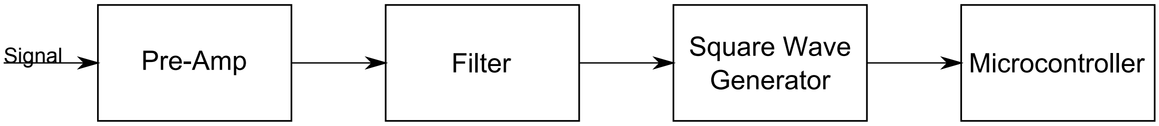


Figure 2: Block Diagram of Analog SubCircuit

4 Timing

In order to calculate the value of a specific cent frequency, the following equation was used:

$$b = a * 2^{\frac{n}{1200}} \tag{1}$$

where b is the desired frequency, a is a lower frequency than b, and n is the number of cents in between a and b. For the positive cent range, we used a = 440Hz and the above formula where n was the number of cents above 440Hz. For the negative cent range, a = 220Hz and n was calculated from

$$n = 1200 - |\text{cent offset}| \tag{2}$$

so if we wanted to find the frequency of -5 cents from A440, n would be 1195. The frequencies found from this formula can be seen in Table 1. Knowing the frequencies corresponding to the cent ranges is useful, but they must be manipulated somewhat for the microcontroller to interpret. The CCP module on the microcontroller measures events by using an internal timer which increments on the instruction clock (Fosc / 4). Therefore, the frequencies calculated in the previous stage need to be converted into a number of instruction cycles. The following equation describes this calculation:

$$(\text{cent frequency})^{-1} * \left(\frac{\text{Instruction Cycles}}{\text{Second}}\right) \tag{3}$$

The converted values calculated from this equation can also be found in Table 1. The Fosc, which is the internal clock frequency, is 16MHz. This means that there are $4 * 106$ instructions per second. We could increase the resolution of our measurements by increasing the clock speed of the microcontroller, but we felt that keeping our design low powered was more important than the increase in resolution gained by using a higher voltage on the microcontroller.

Cents from A440	Frequency (Hz)	Cycles @ Fosc = 16MHz	Lower Cycle Bound	Upper Cycle Bound
-35	431.1939264	9276	9263	TMR
-30	432.4410634	9250	9236	9263
-25	433.6918074	9223	9210	9236
-20	434.946169	9197	9184	9210
-15	436.2041585	9170	9157	9184
-10	437.4657865	9144	9130	9157
-5	438.7310635	9117	9104	9130
0	440	9091	9078	9104
5	441.2726067	9065	9052	9078
10	442.588941	9038	9025	9052
15	443.8288729	9012	8999	9025
20	445.1125537	8986	8974	8999
25	446.399944	8961	8948	8974
30	447.6910645	8935	8922	8948
35	448.985916	8909	1	8922

Table 1: Cent Step Calculations

The lower and upper bounds were calculated by averaging adjacent cycle counts. It is important to note that the cycle counts are whole numbers since it is not possible to have a fractional cycle. All cycle values were rounded to the nearest integer value, which means that the ranges intervals may be off by at most 1-2 cycles, which we deemed as an acceptable imprecision.

Our tuner will take an input signal in the form of a square wave. to be tested. When the input pitch matches A-440Hz, the center LED will light up. As the pitch moves out of tune, other LEDs will light up to signify how sharp or flat the pitch is compared to A. These LED's will be set to 5 cent intervals ranging from -30 to 30 cents around A. Anything above or below these ranges will simply light up the leftmost or rightmost LED depending on whether they are above or below the required pitch. The decision of which LED to light up will be handled by a microcontroller.

5 Microcontroller

We decided to use a microcontroller for the signal processing element of our tuner. Microcontrollers allow quite a bit of flexibility in the processing of the audio signal. We chose the PIC18f4550 partially because this microcontroller had enough ports to drive our LED array. However, the primary reason that this microcontroller was chosen was due to our prior familiarity with the architecture and functionality of the device. Given the limited requirements of the tuner device, there were no special requirements of the microcontroller, which gave us a great degree of flexibility with our choice.

The code for the microcontroller is reliant on three main functionalities of the PIC18f4550: the TMR1 module, the CCP module, and the Primary Idle Mode.

The first module we used is called the TMR1 module, which is a simple timer module. This timer was configured to increment on each instruction cycle to ensure the highest resolution possible. This timer was chosen because it had the ability to hold 16 bit numbers, which was necessary for the number of cycles we needed to count. This timer also works in conjunction with the CCP module, which will be discussed in greater detail later. We used the timer to count the number of cycles for each period of the input signal, but it also has another important function. If the timer were to roll over (meaning that it attempted to increment above $2^1$6), then the program will vector to an interrupt and reset the averaging function used in the CCP interrupt. If this event occurred, then that would mean that the input signal was no longer transmitting, meaning that it would be necessary to clear out the stored averaging values in order to maintain a clear reading the next time an input was detected on the port. More information on the TMR1 module can be found on page 143 of the PIC1f4550 datasheet.

The CCP (Capture, Control , Pulse Width Modulation) Module has a wide variety of uses. For this application, only the capture mode will be used. While configured in the capture mode, the microcontroller waits for a trigger on an event occurring on a specific port of the device. This event, which the user can define as a falling or rising edge transition, will generate an interrupt and vector the CPU of the microcontroller away from the main function. We chose to configure the capture module to copy the contents of the TMR1 counter register into the CCP1REG (a register for use by the CCP module). This transfer is performed automatically by the microcontroller. Basically, this interrupt is generated for every complete cycle of the input signal, meaning that the timer module will record the number of instruction cycles for each period of the signal. The cycle counts corresponding to the cent ranges are shown in Table 1. The cycle count is processed within the interrupt and averaged to ensure that the measurements accurately represent the frequency of the input signal. More information on the CCP modules can be found on page 129 of the PIC18f4550 datasheet.

The final module that our code utilizes is called the Primary Idle Mode. Since the majority of time our code is spent waiting on interrupts to be generated, much of the microcontroller can be disabled during the idle periods. The primary idle is a configuration of the sleep mode available for use on the PIC18f4550. This configuration disables the CPU of the microcontroller, meaning that no more instructions will be processed. Although the CPU is disabled, the peripheral devices, such as the timer modules, will continue to be clocked from the primary clock source. There is a bit of a delay in restarting the microcontroller from the idle mode, but since the TMR1 module is copied over to the CCP1REG by internal hardware, this does not affect the accuracy of our measurements. Also, since our instruction clock was on the order of about 3dB faster than an octave above the desired pitch frequency, we assumed that the wake up time was fairly negligible. We made the same general assumption about the interrupt service routine which calculates the average value. In other words, given the fact that we will be executing approximately 9000 cycles for the desired pitch range, our ISR will not overlap 2 sequential CCP events. More information on the Idle modes can be found on page 40 of the PIC18f4550 datasheet.

Figure 3 represents the basic flow of the microcontroller program. As can be seen from the figure, after the setup and main functions have been run, the microcontroller enters an idle mode. This idle mode is exited when an interrupt is generated. If the interrupt is a CCP interrupt, the average cycle count will be updated. This new average will then be used to decide which LEDs should be turned on. The LEDs are update and program control returns to the main loop. If a TMR1 interrupt is generated, both the average and the display will be reset, and program control will be returned to the main loop.

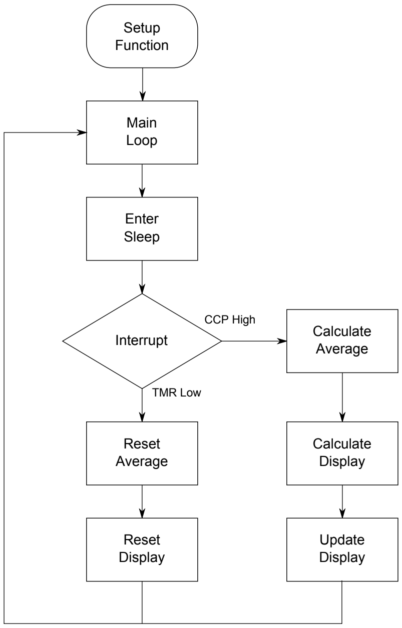


Figure 3: Program Flow Diagram

6 Power Consumption

Power consumption was a big issue for this project because it is meant to be run off of battery power. We will be powering the circuit from a rechargeable lithium polymer battery. Due to stock and size constraints we are limited to two options for capacity, 150 mAh and 3 Ah. Even if we assume the worst case calculations, see Table 2, the 150 mAh battery will power the circuit for more than 3 hours. The real time will be closer to eight hours as we will be spending most of our time with the microcontroller sleeping and all the LEDs won't be constantly on.

Device	Current Con. (Worst Case)	Current Con. (Typical, Active)	Current Con. (Typical, Sleep)	Current Con. (Typical, No Input)
μC [1]	35 mA	16 mA	6.2 mA	6.2 mA
Analog Subcircuit [2]	680 μA	400 μA	400 μA	400 μA
LEDs [3]	7 mA	4 mA	4 mA	1 mA
Power Regulator [4]	50 μA	50 μA	50 μA	50 μA
Total	42 mA	20 mA	11 mA	8 mA

Table 2: Power Consumption

7 Group Member Workload

Patrick Barrett

- Analog Circuit Design
- Analog Simulation
- Schmatic Capture
- Parts Selection
- PCB Layout
- Design Report

Mark Pennebecker

- Microcontroller Selection/Setup
- Programming
- Design Report

Kayiita Johnson

- Programming

Prashant Dhakal

- Simulation

[1] PIC18f4550 Datasheet Pages 375-376

[2] MCP6001 Datasheet

[3] $\frac{3.3V-2.3V}{1K\Omega} = 1\text{mA} *$ N number of LEDs

[4] MCP1703 Datasheet

3.4.1 PRI_IDLE MODE

This mode is unique among the three low-power Idle modes in that it does not disable the primary device clock. For timing sensitive applications, this allows for the fastest resumption of device operation, with its more accurate primary clock source, since the clock source does not have to "warm up" or transition from another oscillator.

PRI_IDLE mode is entered from PRI_RUN mode by setting the IDLEN bit and executing a `SLEEP` instruction. If the device is in another Run mode, set IDLEN first, then clear the SCS bits and execute `SLEEP`. Although the CPU is disabled, the peripherals continue to be clocked from the primary clock source specified by the FOSC3:FOSC0 Configuration bits. The OSTS bit remains set (see Figure 3-7).

When a wake event occurs, the CPU is clocked from the primary clock source. A delay of interval TCSD is required between the wake event and when code execution starts. This is required to allow the CPU to become ready to execute instructions. After the wake-up, the OSTS bit remains set. The IDLEN and SCS bits are not affected by the wake-up (see Figure 3-8).

3.4.2 SEC_IDLE MODE

In SEC_IDLE mode, the CPU is disabled but the peripherals continue to be clocked from the Timer1 oscillator. This mode is entered from SEC_RUN by setting the IDLEN bit and executing a `SLEEP` instruction. If the device is in another Run mode, set IDLEN first, then set SCS1:SCS0 to '`01`' and execute `SLEEP`. When the clock source is switched to the Timer1 oscillator, the primary oscillator is shut down, the OSTS bit is cleared and the T1RUN bit is set.

When a wake event occurs, the peripherals continue to be clocked from the Timer1 oscillator. After an interval of TCSD following the wake event, the CPU begins executing code being clocked by the Timer1 oscillator. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run (see Figure 3-8).

Note:	The Timer1 oscillator should already be running prior to entering SEC_IDLE mode. If the T1OSCEN bit is not set when the `SLEEP` instruction is executed, the `SLEEP` instruction will be ignored and entry to SEC_IDLE mode will not occur. If the Timer1 oscillator is enabled but not yet running, peripheral clocks will be delayed until the oscillator has started. In such situations, initial oscillator operation is far from stable and unpredictable operation may result.

FIGURE 3-7: TRANSITION TIMING FOR ENTRY TO IDLE MODE

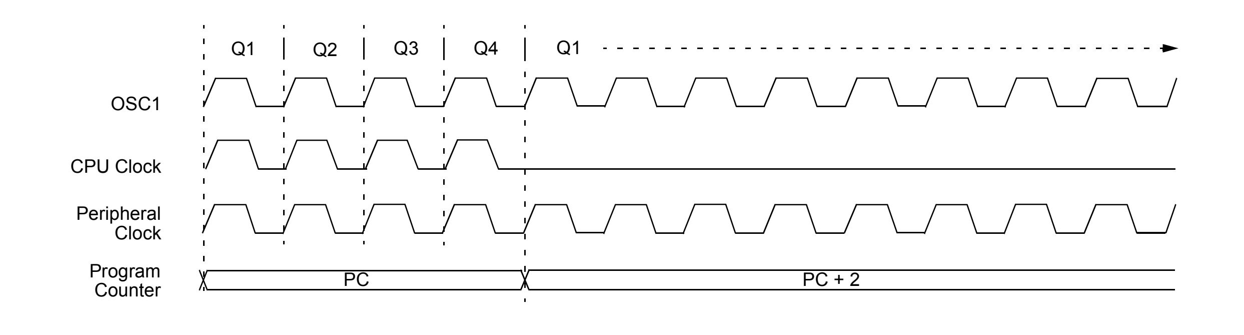
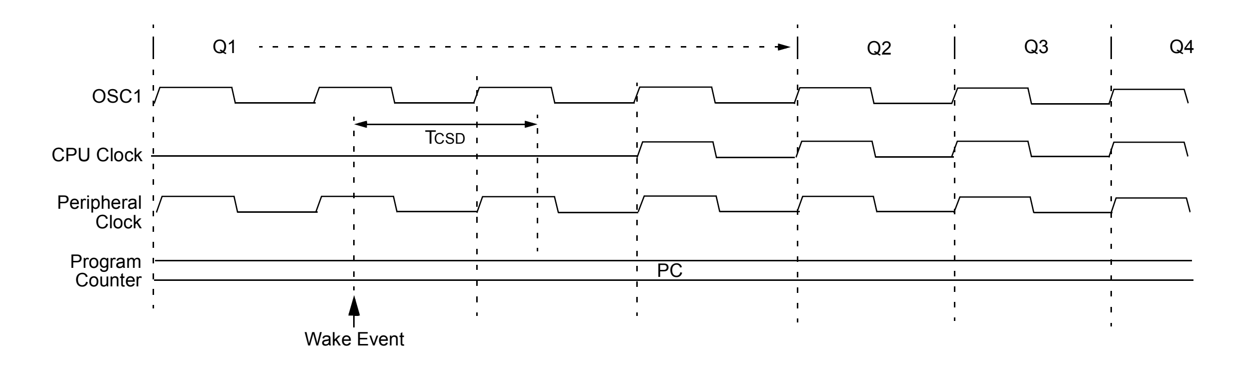


FIGURE 3-8: TRANSITION TIMING FOR WAKE FROM IDLE TO RUN MODE



12.0 TIMER1 MODULE

The Timer1 timer/counter module incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR1H and TMR1L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt on overflow
- Module Reset on CCP Special Event Trigger
- Device clock status flag (T1RUN)

A simplified block diagram of the Timer1 module is shown in Figure 12-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 12-2.

The module incorporates its own low-power oscillator to provide an additional clocking option. The Timer1 oscillator can also be used as a low-power clock source for the microcontroller in power-managed operation.

Timer1 can also be used to provide Real-Time Clock (RTC) functionality to applications with only a minimal addition of external components and code overhead.

Timer1 is controlled through the T1CON Control register (Register 12-1). It also contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

REGISTER 12-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

bit 6 **T1RUN:** Timer1 System Clock Status bit
1 = Device clock is derived from Timer1 oscillator
0 = Device clock is derived from another source

bit 5-4 **T1CKPS1:T1CKPS0:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Select bit
When TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from RC0/T1OSO/T13CKI pin (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

15.0 CAPTURE/COMPARE/PWM (CCP) MODULES

PIC18F2455/2550/4455/4550 devices all have two CCP (Capture/Compare/PWM) modules. Each module contains a 16-bit register, which can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register.

In 28-pin devices, the two standard CCP modules (CCP1 and CCP2) operate as described in this chapter. In 40/44-pin devices, CCP1 is implemented as an Enhanced CCP module, with standard Capture and Compare modes and Enhanced PWM modes. The ECCP implementation is discussed in **Section 16.0 "Enhanced Capture/Compare/PWM (ECCP) Module"**.

The Capture and Compare operations described in this chapter apply to all standard and Enhanced CCP modules.

> **Note:** Throughout this section and **Section 16.0 "Enhanced Capture/Compare/PWM (ECCP) Module"**, references to the register and bit names for CCP modules are referred to generically by the use of 'x' or 'y' in place of the specific module number. Thus, "CCPxCON" might refer to the control register for CCP1, CCP2 or ECCP1. "CCPxCON" is used throughout these sections to refer to the module control register regardless of whether the CCP module is a standard or Enhanced implementation.

REGISTER 15-1: CCPxCON: STANDARD CCPx CONTROL REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—[1]	—[1]	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'[1]

bit 5-4 **DCxB1:DCxB0**: PWM Duty Cycle Bit 1 and Bit 0 for CCPx Module

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight MSbs of the duty cycle are found in CCPR1L.

bit 3-0 **CCPxM3:CCPxM0**: CCPx Module Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0001 = Reserved
0010 = Compare mode: toggle output on match (CCPxIF bit is set)
0011 = Reserved
0100 = Capture mode: every falling edge
0101 = Capture mode: every rising edge
0110 = Capture mode: every 4th rising edge
0111 = Capture mode: every 16th rising edge
1000 = Compare mode: initialize CCPx pin low; on compare match, force CCPx pin high (CCPxIF bit is set)
1001 = Compare mode: initialize CCPx pin high; on compare match, force CCPx pin low (CCPxIF bit is set)
1010 = Compare mode: generate software interrupt on compare match (CCPxIF bit is set, CCPx pin reflects I/O state)
1011 = Compare mode: trigger special event, reset timer, start A/D conversion on CCPx match (CCPxIF bit is set)
11xx = PWM mode

Note 1: These bits are not implemented on 28-pin devices and are read as '0'.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F2455/2550/4455/4550 (Industrial) PIC18LF2455/2550/4455/4550 (Industrial) (Continued)

PIC18LF2455/2550/4455/4550 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC18F2455/2550/4455/4550 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial

Param No.	Symbol	Device	Typ	Max	Units	Conditions		
		Supply Current (IDD)(2)						
		PIC18LFX455/X550	250	500	μA	-40°C	VDD = 2.0V	FOSC = 1 MHZ (**PRI_RUN**, EC oscillator)
			250	500	μA	+25°C		
			250	500	μA	+85°C		
		PIC18LFX455/X550	550	650	μA	-40°C	VDD = 3.0V	
			480	650	μA	+25°C		
			460	650	μA	+85°C		
		All devices	1.2	1.6	mA	-40°C	VDD = 5.0V	
			1.1	1.5	mA	+25°C		
			1.0	1.4	mA	+85°C		
		PIC18LFX455/X550	0.74	2.0	mA	-40°C	VDD = 2.0V	FOSC = 4 MHz (**PRI_RUN**, EC oscillator)
			0.74	2.0	mA	+25°C		
			0.74	2.0	mA	+85°C		
		PIC18LFX455/X550	1.3	3.0	mA	-40°C	VDD = 3.0V	
			1.3	3.0	mA	+25°C		
			1.3	3.0	mA	+85°C		
		All devices	2.7	6.0	mA	-40°C	VDD = 5.0V	
			2.6	6.0	mA	+25°C		
			2.5	6.0	mA	+85°C		
		All devices	15	35	mA	-40°C	VDD = 4.2V	FOSC = 40 MHZ (**PRI_RUN**, EC oscillator)
			16	35	mA	+25°C		
			16	35	mA	+85°C		
		All devices	21	40	mA	-40°C	VDD = 5.0V	
			21	40	mA	+25°C		
			21	40	mA	+85°C		
		All devices	20	40	mA	-40°C	VDD = 4.2V	FOSC = 48 MHZ (**PRI_RUN**, EC oscillator)
			20	40	mA	+25°C		
			20	40	mA	+85°C		
		All devices	25	50	mA	-40°C	VDD = 5.0V	
			25	50	mA	+25°C		
			25	50	mA	+85°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
$\overline{\text{OSC1}}$ = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

3: Standard low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F2455/2550/4455/4550 (Industrial) PIC18LF2455/2550/4455/4550 (Industrial) (Continued)

PIC18LF2455/2550/4455/4550 (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC18F2455/2550/4455/4550 (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Symbol**	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
		Supply Current (IDD)[2]						
		PIC18LFX455/X550	65	130	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz (**PRI_IDLE** mode, EC oscillator)
			65	120	μA	+25°C		
			70	115	μA	+85°C		
		PIC18LFX455/X550	120	270	μA	-40°C	VDD = 3.0V	
			120	250	μA	+25°C		
			130	240	μA	+85°C		
		All devices	230	480	μA	-40°C	VDD = 5.0V	
			240	450	μA	+25°C		
			250	430	μA	+85°C		
		PIC18LFX455/X550	255	475	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz (**PRI_IDLE** mode, EC oscillator)
			260	450	μA	+25°C		
			270	430	μA	+85°C		
		PIC18LFX455/X550	420	900	μA	-40°C	VDD = 3.0V	
			430	850	μA	+25°C		
			450	810	μA	+85°C		
		All devices	0.9	1.5	mA	-40°C	VDD = 5.0V	
			0.9	1.4	mA	+25°C		
			0.9	1.3	mA	+85°C		
		All devices	6.0	16	mA	-40°C	VDD = 4.2V	FOSC = 40 MHz (**PRI_IDLE** mode, EC oscillator)
			6.2	16	mA	+25°C		
			6.6	16	mA	+85°C		
		All devices	8.1	18	mA	-40°C	VDD = 5.0V	
			8.3	18	mA	+25°C		
			9.0	18	mA	+85°C		
		All devices	8.0	18	mA	-40°C	VDD = 4.2V	FOSC = 48 MHz (**PRI_IDLE** mode, EC oscillator)
			8.1	18	mA	+25°C		
			8.2	18	mA	+85°C		
		All devices	9.8	21	mA	-40°C	VDD = 5.0V	
			10.0	21	mA	+25°C		
			10.5	21	mA	+85°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

3: Standard low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

1.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

$V_{DD} – V_{SS}$	7.0V
Current at Analog Input Pins (V_{IN}+, V_{IN}–)	±2 mA
Analog Inputs (V_{IN}+, V_{IN}–) ††	V_{SS} – 1.0V to V_{DD} + 1.0V
All Other Inputs and Outputs	V_{SS} – 0.3V to V_{DD} + 0.3V
Difference Input Voltage	$|V_{DD} – V_{SS}|$
Output Short Circuit Current	Continuous
Current at Output and Supply Pins	±30 mA
Storage Temperature	–65°C to +150°C
Maximum Junction Temperature (T_J)	+150°C
ESD Protection On All Pins (HBM; MM)	≥ 4 kV; 200V

† Notice: Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operational listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

†† See **Section 4.1.2 “Input Voltage and Current Limits”**.

DC ELECTRICAL SPECIFICATIONS

Electrical Characteristics: Unless otherwise indicated, T_A = +25°C, V_{DD} = +1.8V to +5.5V, V_{SS} = GND, $V_{CM} = V_{DD}/2$, $V_L = V_{DD}/2$, R_L = 10 kΩ to V_L, and $V_{OUT} \approx V_{DD}/2$ (refer to Figure 1-1).

Parameters	Sym	Min	Typ	Max	Units	Conditions
Input Offset						
Input Offset Voltage	V_{OS}	-4.5	—	+4.5	mV	$V_{CM} = V_{SS}$ **(Note 1)**
Input Offset Drift with Temperature	$\Delta V_{OS}/\Delta T_A$	—	±2.0	—	µV/°C	T_A= -40°C to +125°C, $V_{CM} = V_{SS}$
Power Supply Rejection Ratio	PSRR	—	86	—	dB	$V_{CM} = V_{SS}$
Input Bias Current and Impedance						
Input Bias Current:	I_B	—	±1.0	—	pA	
Industrial Temperature	I_B	—	19	—	pA	T_A = +85°C
Extended Temperature	I_B	—	1100	—	pA	T_A = +125°C
Input Offset Current	I_{OS}	—	±1.0	—	pA	
Common Mode Input Impedance	Z_{CM}	—	10^{13}||6	—	Ω||pF	
Differential Input Impedance	Z_{DIFF}	—	10^{13}||3	—	Ω||pF	
Common Mode						
Common Mode Input Range	V_{CMR}	V_{SS} – 0.3	—	V_{DD} + 0.3	V	
Common Mode Rejection Ratio	CMRR	60	76	—	dB	V_{CM} = -0.3V to 5.3V, V_{DD} = 5V
Open-Loop Gain						
DC Open-Loop Gain (Large Signal)	A_{OL}	88	112	—	dB	V_{OUT} = 0.3V to V_{DD} – 0.3V, $V_{CM} = V_{SS}$
Output						
Maximum Output Voltage Swing	V_{OL}, V_{OH}	V_{SS} + 25	—	V_{DD} – 25	mV	V_{DD} = 5.5V, 0.5V Input Overdrive
Output Short Circuit Current	I_{SC}	—	±6	—	mA	V_{DD} = 1.8V
		—	±23	—	mA	V_{DD} = 5.5V
Power Supply						
Supply Voltage	V_{DD}	1.8	—	6.0	V	**Note 2**
Quiescent Current per Amplifier	I_Q	50	100	170	µA	I_O = 0, V_{DD} = 5.5V, V_{CM} = 5V

Note 1: MCP6001/1R/1U/2/4 parts with date codes prior to December 2004 (week code 49) were tested to ±7 mV minimum/maximum limits.

2: All parts with date codes November 2007 and later have been screened to ensure operation at V_{DD} = 6.0V. However, the other minimum and maximum specifications are measured at 1.8V and 5.5V.

MCP1703

250 mA, 16V, Low Quiescent Current LDO Regulator

Features:

- 2.0 µA Typical Quiescent Current
- Input Operating Voltage Range: 2.7V to16.0V
- 250 mA Output Current for Output Voltages $\geq$ 2.5V
- 200 mA Output Current for Output Voltages < 2.5V
- Low Dropout Voltage, 625 mV typical @ 250 mA for V_R = 2.8V
- 0.4% Typical Output Voltage Tolerance
- Standard Output Voltage Options:
  - 1.2V, 1.5V, 1.8V, 2.5V, 2.8V, 3.0V, 3.3V, 4.0V, 5.0V
- Output Voltage Range: 1.2V to 5.5V in 0.1V Increments (50 mV increments available upon request)
- Stable with 1.0 µF to 22 µF Ceramic Output Capacitance
- Short-Circuit Protection
- Overtemperature Protection

Applications:

- Battery-Powered Devices
- Battery-Powered Alarm Circuits
- Smoke Detectors
- CO^2 Detectors
- Pagers and Cellular Phones
- Smart Battery Packs
- Low Quiescent Current Voltage Reference
- PDAs
- Digital Cameras
- Microcontroller Power
- Solar-Powered Instruments
- Consumer Products
- Battery-Powered Data Loggers

Related Literature:

- AN765, "*Using Microchip's Micropower LDOs*", DS00765, Microchip Technology Inc., 2002
- AN766, "*Pin-Compatible CMOS Upgrades to Bipolar LDOs*", DS00766, Microchip Technology Inc., 2002
- AN792, "*A Method to Determine How Much Power a SOT23 Can Dissipate in an Application*", DS00792, Microchip Technology Inc., 2001

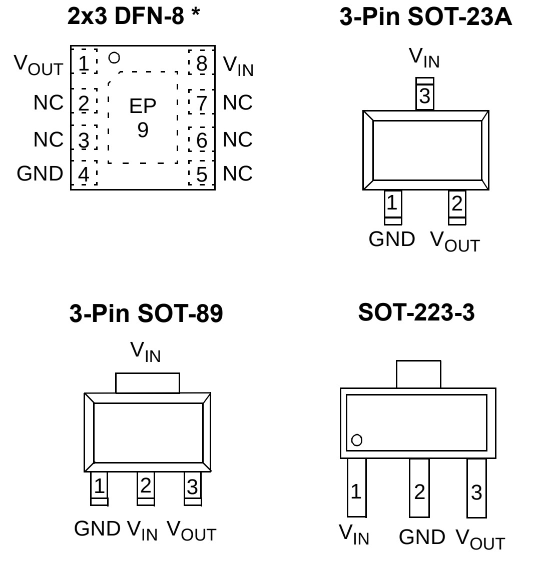
Description:

The MCP1703 is a family of CMOS low dropout (LDO) voltage regulators that can deliver up to 250 mA of current while consuming only 2.0 µA of quiescent current (typical). The input operating range is specified from 2.7V to 16.0V, making it an ideal choice for two to six primary cell battery-powered applications, 9V alkaline and one or two cell Li-Ion-powered applications.

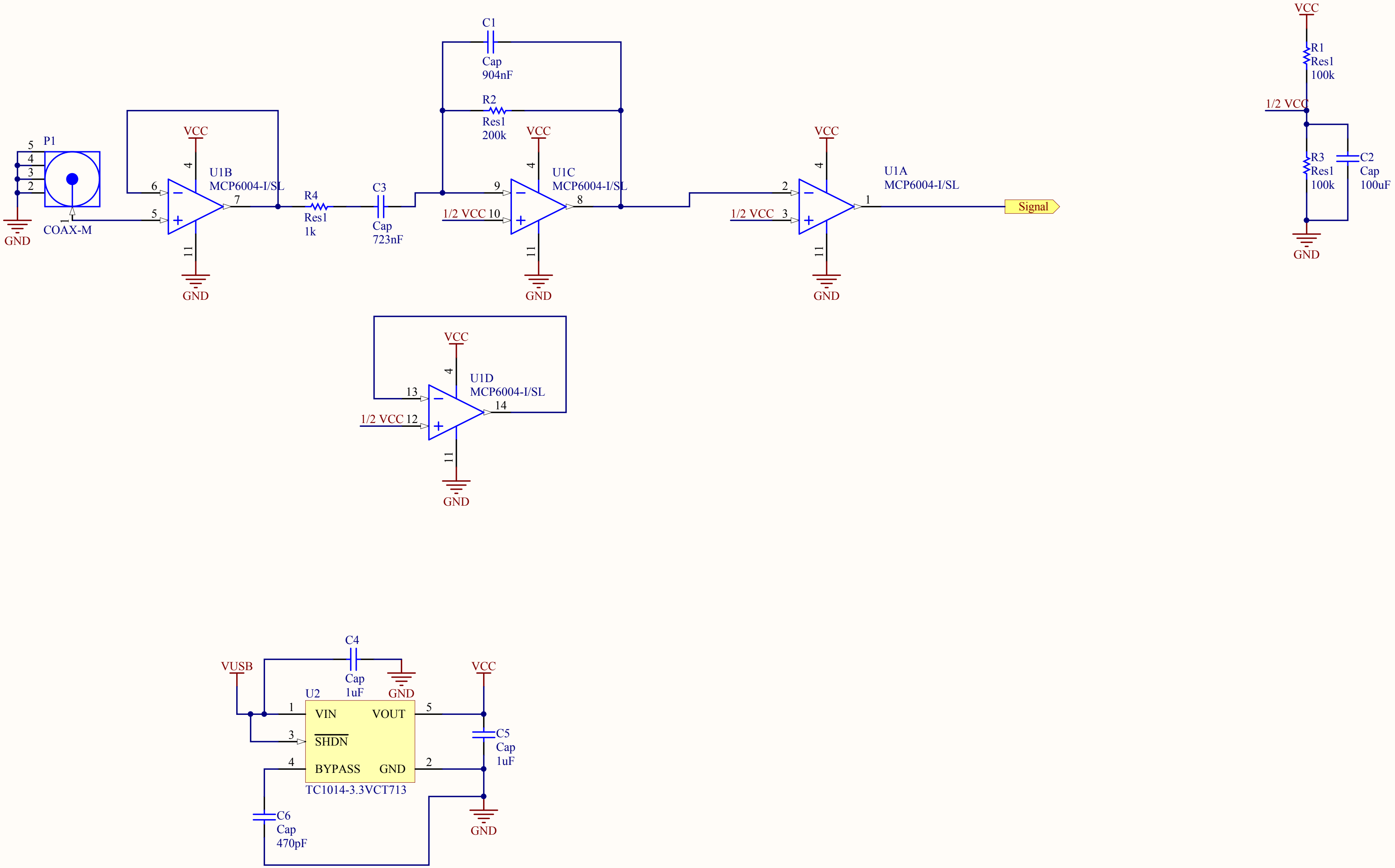
The MCP1703 is capable of delivering 250 mA with only 625 mV (typical) of input to output voltage differential (V_{OUT} = 2.8V). The output voltage tolerance of the MCP1703 is typically ±0.4% at +25°C and ±3% maximum over the operating junction temperature range of -40°C to +125°C. Line regulation is ±0.1% typical at +25°C.

Output voltages available for the MCP1703 range from 1.2V to 5.5V. The LDO output is stable when using only 1 µF of output capacitance. Ceramic, tantalum, or aluminum electrolytic capacitors can all be used for input and output. Overcurrent limit and overtemperature shutdown provide a robust solution for any application. Package options include the SOT-223-3, SOT-23A, 2x3 DFN-8, and SOT-89-3.

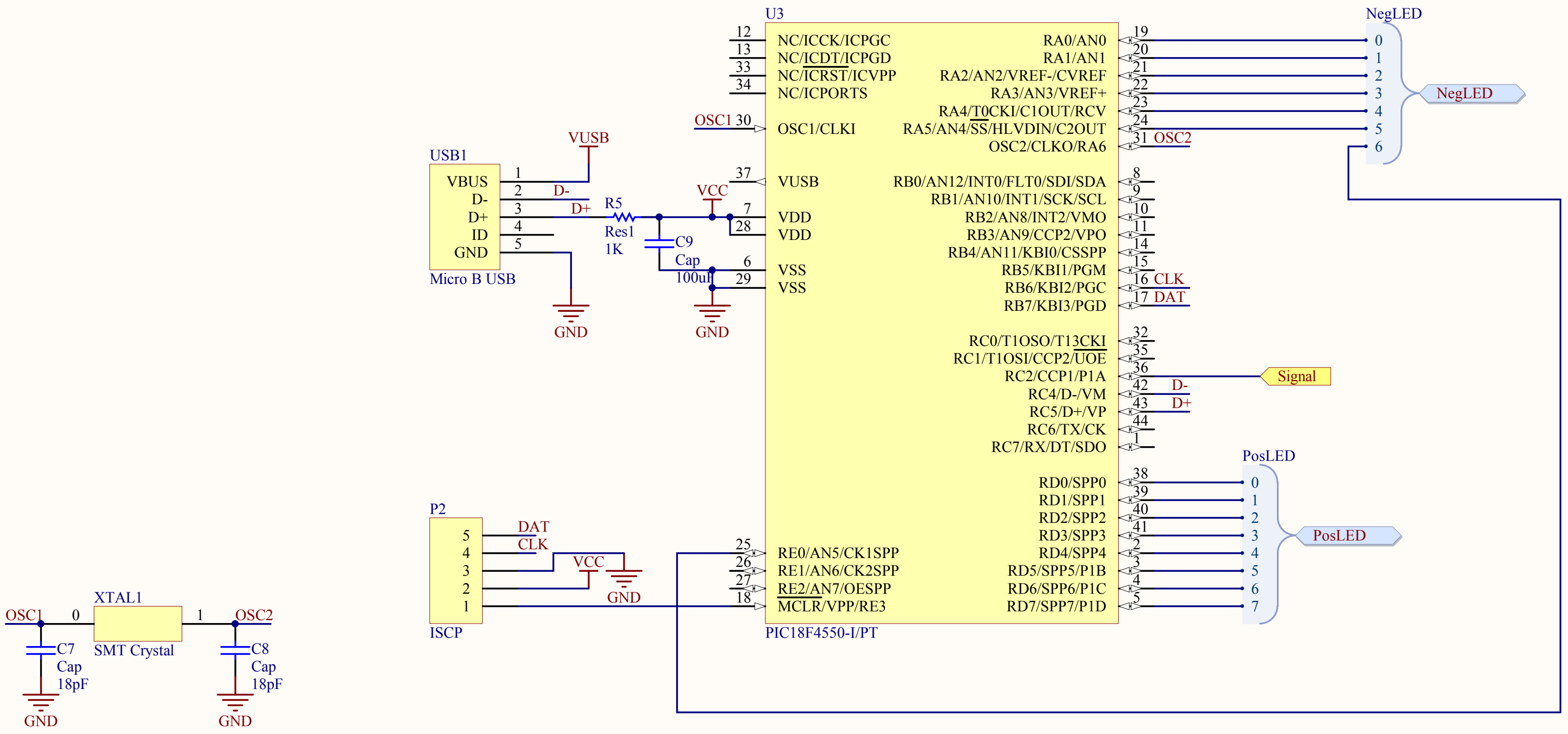
Package Types



* Includes Exposed Thermal Pad (EP); see Table 3-1.



Title		
Size Letter	Number	Revision
Date: 3/7/2012		Sheet of
File: C:\Users\..\Analog.SchDoc		Drawn By:



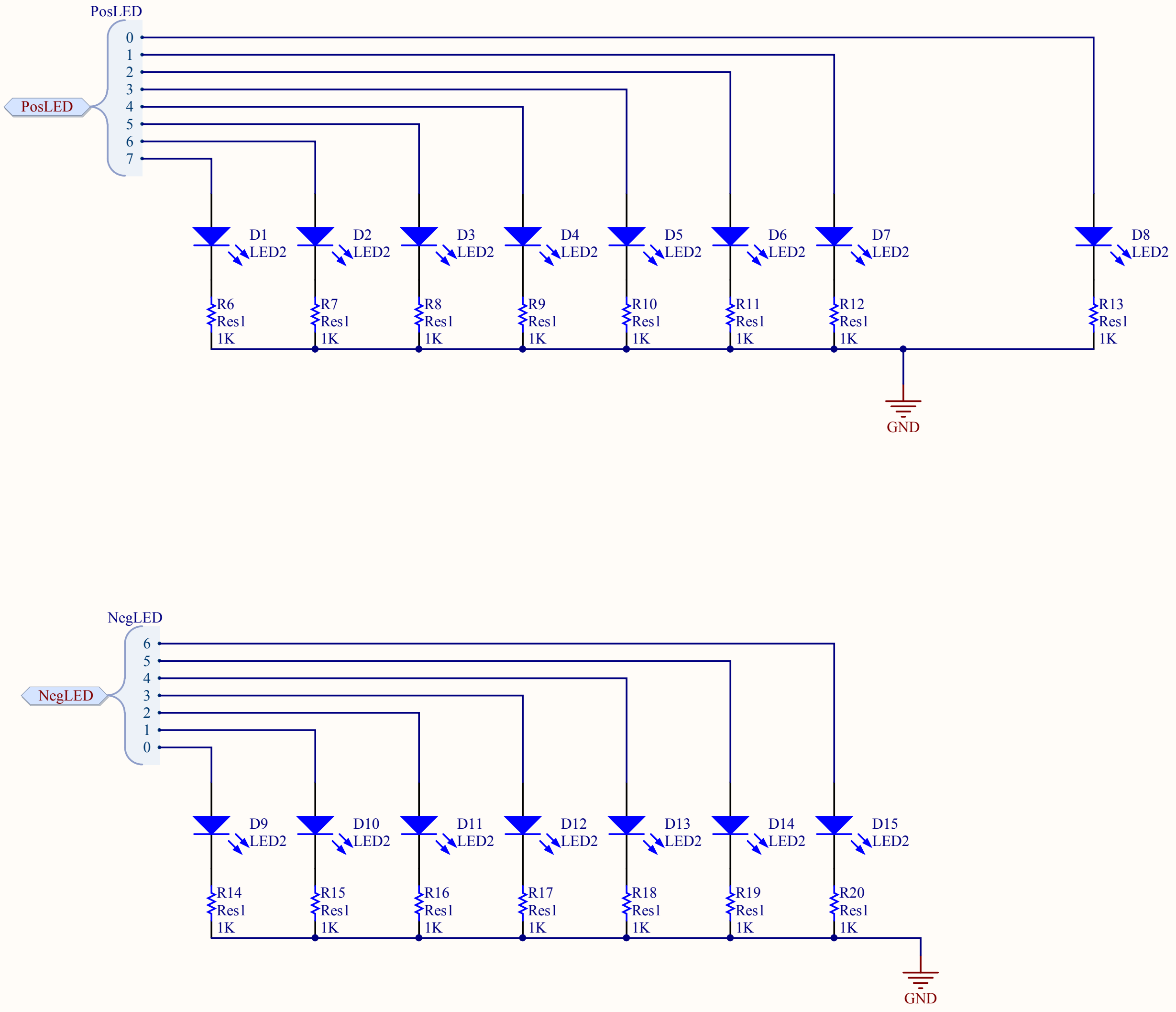
Title

Size
Letter

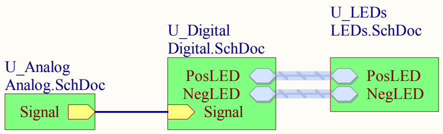
Number

Revision

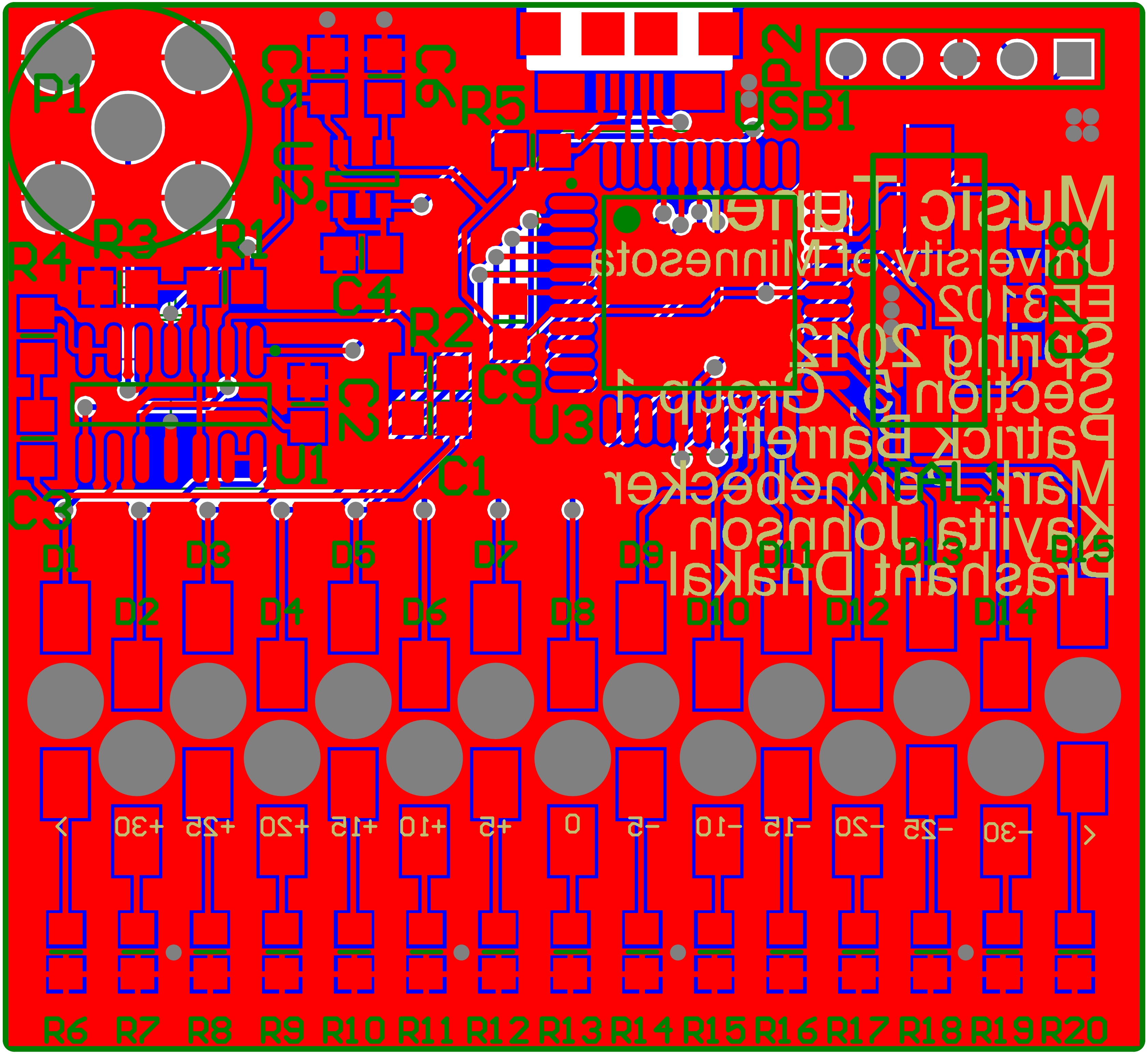
Date: 3/7/2012
Sheet of
File: C:\Users\..\Digital.SchDoc
Drawn By:



Title			
Size Letter	Number		Revision
Date:	3/7/2012	Sheet of	
File:	C:\Users\..\LEDs.SchDoc	Drawn By:	



Title		
Size Letter	Number	Revision
Date: 3/7/2012		Sheet of
File: C:\Users\..\Main Board.SchDoc		Drawn By:


P1
C5
C6
R5
P2
USB1
U2
R3
R1
R4
C4
R2
C9
U3
C2
U1
C1
C3
C8
C7
XTAL1
Music Tuner
University of Minnesota
EE3102
Spring 2012
Section 5, Group 1
Patrick Barrett
Mark Kurnebecker
Kavita Johnson
Prashant Dhakal
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
D14
D15
<
+30
+25
+20
+15
+10
+5
0
-5
-10
-15
-20
-25
-30
>
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
R16
R17
R18
R19
R20