ColourSense Inc. Reveals True Colours

March 11th, 2005

Lucky One
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

RE: ENSC 440 Design Specification for a Colour Identifying Device

Dear Mr. One:

The enclosed document, *Design Specifications for a Colour Identifying Device—ColourIt*, explains the methods of design and choices in implementation.

We are in the process of developing a handheld device that can determine the colour of a point. Capable of distinguishing 147 colours, it is ideal for designers, painters, hobbyists, as well as the colour vision deficient.

The purpose of this functional specification is to ensure our project will meet the specific needs of the user. We will use this set of clearly defined requirements of the system as a guideline for our final product.

ColourSense Inc. is comprised of three innovative, diligent and ambitious fourth year engineering students – Julie Chao, Linda Ni and Bill Yang. I am more than willing to answer any questions or concerns in regards to our functional specification. Please feel free to contact me by phone at 604-764-2608 or by email at coloursense-ensc440@sfu.ca.

Sincerely,

Julie Chao
Chief Executive Officer
ColourSense Inc.

Enclosure: *Design Specification for a Colour Identifying Device—ColourIt*

ColourSense Inc.

Proposal for a Colour Identifying Device

ColourIt

Team Members:	Julie Chao	ctchao@sfu.ca
	Linda Ni	lni@sfu.ca
	Bill Yang	billy@sfu.ca
Submitted to:	Lucky One	lucky@sfu.ca
	Amir Niroumand	amniroum@sfu.ca
	Mike Sjoerdsma	msjoerds@sfu.ca
	Tony Ottaviani	tottavia@sfu.ca
	Scott Logie	slogie@sfu.ca
	Steve Whitmore	whitmore@cs.sfu.ca

Issued Date: March 11th, 2005

Executive Summary

In John Locke' s famous essay of 1690, he pondered whether the "*violet produced in one man's mind by his eyes were the same that a marigold produced in another man's, and vice versa*[1]." The colours perceived by an individual would not be the same as the colours perceived by another at a different time or different place. While philosophers and scientists have debated this issue for centuries, the engineers of ColourSense have endeavoured to resolve it through the development of a portable colour-identifying device - ColourIt.

ColourIt is a handy device capable of recognizing 147 colours defined by the World Wide Web Consortium [1]. ColourIt is shaped like a flashlight, with a LCD screen attached on the side to inform the user the name of the colour and the levels of red, green and blue it is composed of. The device will be portable, affordable and user-friendly.

The unending list of applications for ColourIt includes colour distinction for the colour vision deficient, exact colour matching and verification for painters and designers, determining color differences between digital prints and the original item, copyright protection of images and detection of art fraud.

In this design specification, we will explain design principles divided into a colour-sensing unit, a central control unit and a user interface unit.

Table of Contents

List of Figures

List of Tables

Glossary

UDATA	user define variables and allocate a byte to each variable
EEPROM	nonvolatile memory where writing to memory is complicated and safeguarded so data stored in that section is not easily overwritten

1. Introduction

ColourIt is a handy device which tells a user the colour of the point he or she is looking at. Shaped like a flashlight, the user aims the device at desired point. Immediately, the name of the colour and the levels of red, green and blue it is composed of are displayed on the LCD screen. The device is capable of recognizing 147 colours defined by the World Wide Web Consortium [1]. When not in use, ColourIt slips easily into a pocket or a purse.

ColourIt is intended for amateurs and professionals in various industries. ColourIt is also will also help those with colour vision deficiency.

1.1. Scope

This design specification is a detailed description of the development of a colour identifying device. The overall system specifications include the colour-sensing unit, the central control unit and the user interface unit. The colour-sensing unit will describe the physical design, electronic design, colour sensor specifications and light source specifications. The central control unit will explain the choice and use of the microcontroller, the control software and the software test plan. Finally, the user interface unit will detail hardware and software.

1.2. Intended Audience

The design specification for ColourIt is primarily intended for the design engineers and the project manager. Designer engineers will develop the device in accordance to this detailed set of system and product requirements. The project manager will use this document to ensure project development and performance objectives are met.

1.3. Acronyms

ICD In Circuit Debugger
LCD Liquid Crystal Display
LED Light Emitting Diode
RGB Red, Green, Blue

2. System Overview

The development of ColourIt is split up into three units: colour sensing, central control and user interface.

Since the physical outcome requires the device to be packaged in a portable and compact manner, the system must be powered by batteries. The device will be able to output the name of the colour on a liquid crystal display (LCD); there are 147 possible colours. The device will have a power switch and two buttons for the user: sample and toggle. The sample button is also the power button; when selected, it detects the colour of the area and outputs the name of the colour on the LCD. The LCD can also output another piece of information for the user; the user can see the pixel numbers that the colour detected is composed of. The toggle button allows the user to switch back and forth between displaying the name of the colour and its pixel numbers. Refer to Figure1 for the integrated system of a working prototype.

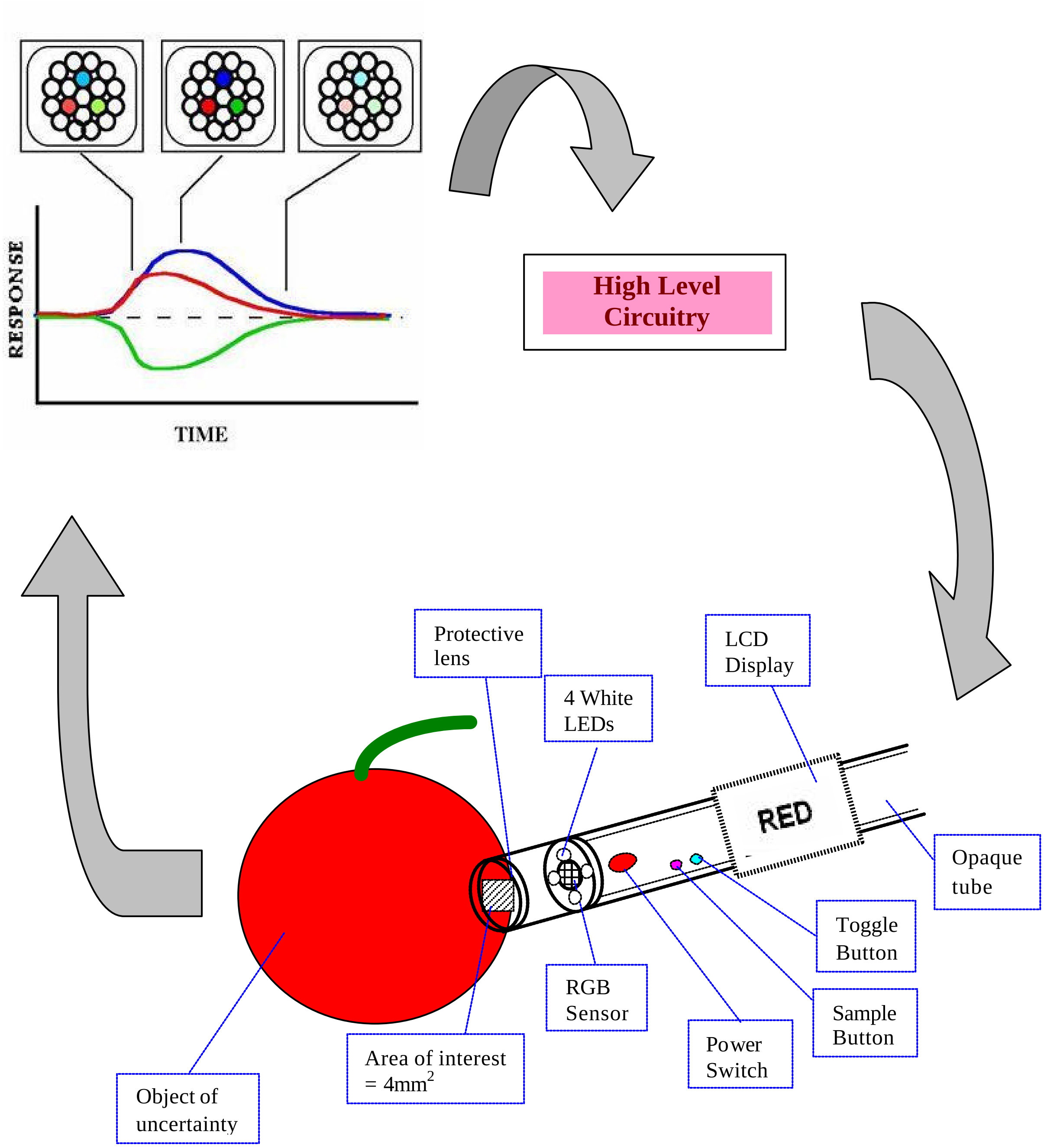


Figure 1: Integrated System for a Working Prototype

3. Colour-Sensing Unit

3.1. Overview

In colour sensing, it is imperative to have consistent colour detection. Hence, the device must isolate external light and provide its own light source to illuminate the area of interest. A colour sensor will pick up the light intensity based on the colour of the area; one single output channel is sent to the microcontroller to be processed for the levels of red, green, blue and intensity.

3.2. Light Source Specifications

For consistent colour sampling, the system must have an invariant light source that is undisturbed by external light.

White LEDs are the best choice for light source because white is composed of red, blue and green. The microcontroller will be providing power for the white LEDs and is capable of outputting up to 5V DC relative to ground. The LEDs cannot require power above this specification and they cannot draw too much current. However, they should be powerful enough to light up the area of interest for the sensor to detect the colour.

After deliberation between trade-offs between each white LED model, the SSL-LX5093XUWC from Lumex is the final choice. The specifications are stated as below:

Table 1: Electrical Characteristics of the White LED

Parameter	Specification
V_f	3.6V
I_f	30mA
I_o	2300mcd
Viewing angle	15°

Four LEDs are chosen to surround the colour sensor to ensure even distribution of the light in the enclosed area (see the following diagram):

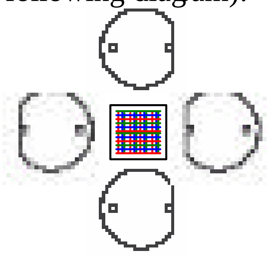


Figure 3: Arrangement of Colour Sensor and White LEDs

1k? resisters are connected to the four anodes to limit the current into the white LEDs. To reduce the number of connections, all four LEDs are grounded together and are paired up so that it only requires two 5V supply lines from the microcontroller.

3.3. Colour Sensor Specifications

The colour sensor chosen is the TCS230 from TAOS, a colour light to frequency converter which combines an array of configurable color filtered silicon photodiodes and a current-to-frequency converter on a single monolithic CMOS integrated circuit [2]. The pin layout of the chip with its centred photodiode array is shown in the following diagram:

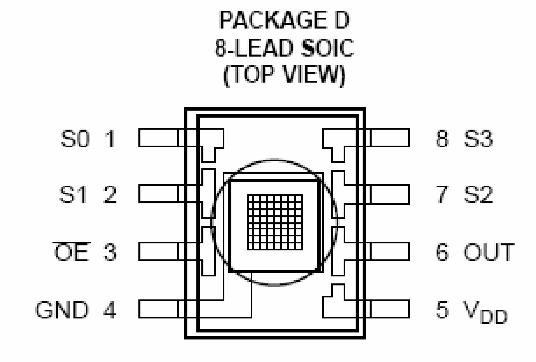


Figure 4: TCS230 RGB Sensor Pin Layout

The TCS230 has an array of 8×8 photodetectors, composed of red, green, blue, and ‘clear’ filters distributed evenly throughout the array to eliminate location bias among the colors. The photodetectors array receives incoming light: white LED light reflected from the surface of the area of interest. The device has an internal oscillator which produces a square-wave (50% duty cycle) output whose frequency is directly proportional to the light intensity incident on the red, green, blue, or unfiltered channel. A block diagram of the sensor system is shown in the following diagram:

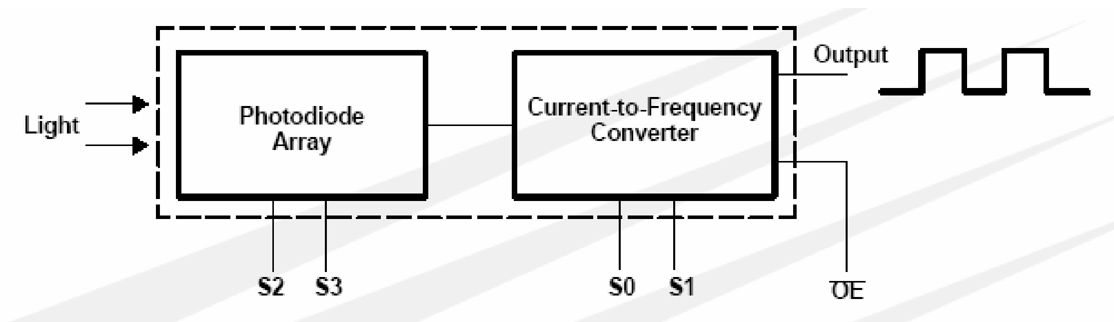


Figure 5: TCS230Functional Block Diagram

The oscillator has a tri-stable output, and the color to be read is selected using address lines S_2 and S_3. In addition, it is possible to program the divide rate of the oscillator using two additional lines, S_0 and S_1. An internal multiplexer allo ws selection of any of the four channels to be sent to the output pin. The settings for these control lines and their functions are summarized below:

The functions of the 8 pins are summarized below:

Table 2: Terminal Functions

TERMINAL NAME	TERMINAL NO.	I/O	DESCRIPTION
GND	4		Power supply ground. All voltages are referenced to GND.
$\overline{OE}$	3	I	Enable for f_o (active low).
OUT	6	O	Output frequency (f_o).
S0, S1	1, 2	I	Output frequency scaling selection inputs.
S2, S3	7, 8	I	Photodiode type selection inputs.
V_{DD}	5		Supply voltage

Pins S_0 and S_1 are used to scale the output frequency to either power down, 2%, 20% or 100% while pins S_2 and S_3 are used to select the colour filter, through the basic logic tables shown below:

Table 3: Terminal Functions

S0	S1	Divide
0	0	Pwr. Down
0	1	1:50
1	0	1:5
1	1	1:1

S2	S3	Color
0	0	Red
0	1	Blue
1	0	Clear
1	1	Green

The main advantage of using the TCS230 over other RGB sensors is because it is programmable and has a direct single wire interface to a microcontroller without the use of a diffuser. The red, green, blue and clear sensors are all integrated in one package and is easily converted to digital format. Although packaged at only 5mm × 6mm footprint, it is capable of achieving a 16-bit resolution. The following figure shows the difference between the approach of typical RGB sensors and the TCS230:

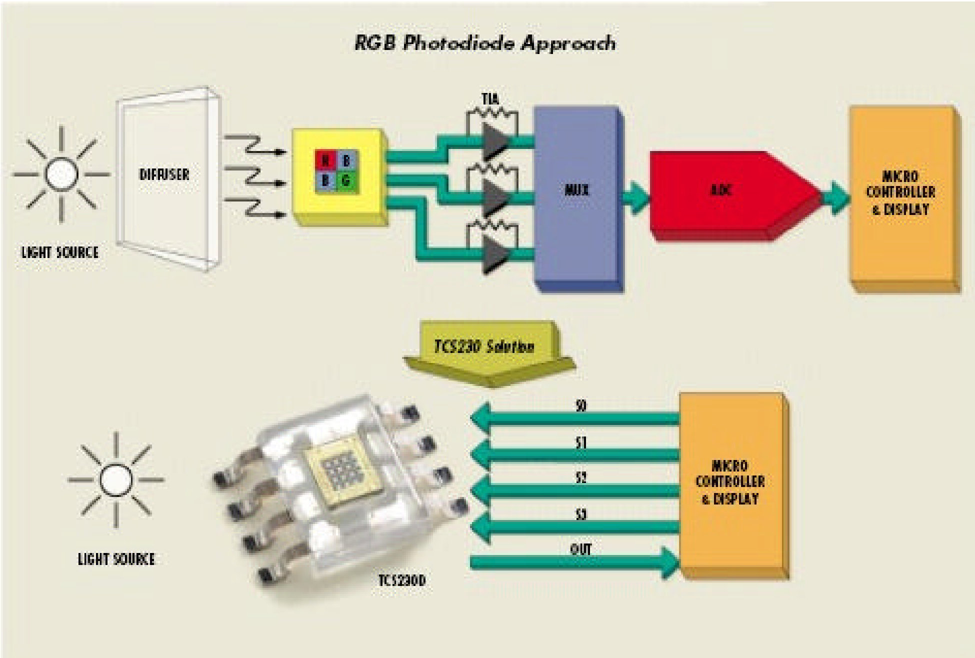


Figure 6: Typical RGB Photodiode and TCS230 Implementation

The microcontroller provides the sensor $V_{DD} = 5V$ and can be directly connected with the addition of a 0.1µF capacitor with short leads mounted close to the device package to decouple the power-supply lines.

3.4. Physical Design

The physical design takes into consideration of the layout of the white LEDs and sensor, with the desire of minimizing the entire setup to increase portability. A hollow casing must be designed so that it can enclose the LEDs and block out nuisance lighting that would influence the reading for the sensor. A cross-sectional area for the hollow casing must first be determined. Refer back to Figure 3 for the optimal arrangement of the four white LEDs and sensor. The diameter of each white LED is 5.6mm in diameter (see Figure 7) and is located on each of the four sides of the sensor.

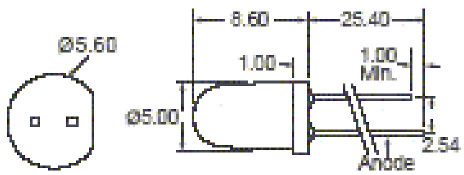


Figure 7: White LED Dimensions

With the dimensions of the sensor (shown in the following figure) taken to be close to 5mm × 6mm, a prototype board area of at least 16.2mm × 17.2mm is required.

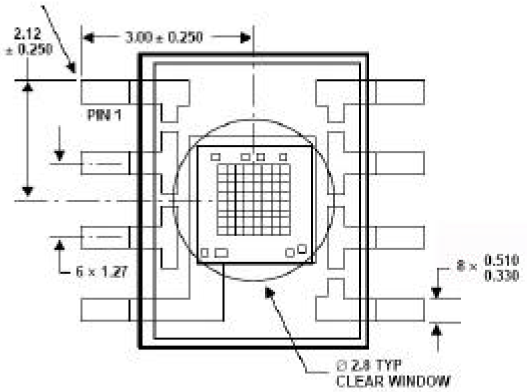


Figure 8: Sensor Dimensions

After providing spacing for the components and wiring, an area of 20mm × 20mm is the final choice of cross-sectional area (refer to Figure 9).

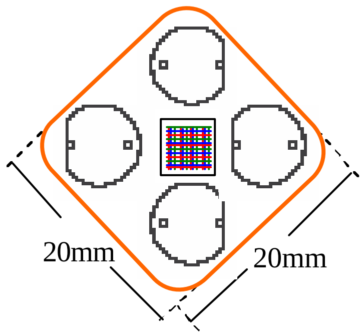


Figure 9: Cross-Sectional Layout of Sensor and LEDs

Because the length of each white LED is 8.6mm and the protoboard is 2mm in thickness, the height of this hollow spacing must be at least 13mm (providing 2.4mm for spacing).

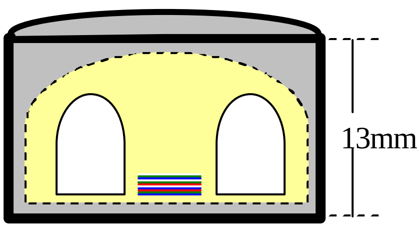


Figure 10: Side View of Colour-Sensing Unit

4. Central Control Unit

4.1. Overview

The central control unit is a bridge between the colour sensing unit and user interface unit. This middle unit revolves the use of control software to handle the microcontroller. The microcontroller is capable of powering and controlling the input signals to the sensor. The oscillatory clock on the microcontroller must be fast enough to receive input from the sensor and able to check the status of two buttons continuously. The data collected from the colour-sensing unit will be processed and prepared for output in the LCD screen in the central control unit.

4.2. Microchip PIC18F452 Microcontroller

4.2.1. Selection Criteria

We considered many options during the microcontroller selection process. The criteria applied for microcontroller selection included:

1. Cost of microcontroller and sample availability
2. Amount of memory available for program and variables
3. Numbers of pins available for I/O
4. Interrupt Capabilities and Clock Speed
5. Cost and availability of Development Tool and Emulator
6. Development Learning Curve

The PIC18FXXX is Microchip's latest developed family. They are relatively attractive to use because of the following features [3]:

- Speed: CPU can execute most of its instructions in 0.1 μs with its 10MHz internal clock and 16 bit bus.
- Flexibly timer resources: 4 independent timers (see Appendix B: PIC18XXX) and 2 capture/compare modules support timing measurements.
- Interrupt control: 17 independent interrupt sources control when CPU deals with each source.
- Robustness: I/O pins can drive loads of up to 25 mA as outputs and are protected against static electricity damage as inputs.
- Free Software Tools: Microchip also provides a free development tool kit, MPLAB IDE, and an assembler, MPASM on their website. The debugging tool, in-circuit-debugger (ICD2), although is not free, could be borrowed from Dr. Andrew Rawicz.

PIC18F452 distinguishes from its family on the basis of I/O pin, memory quantity and low-power operation (see Appendix B). In addition to being an exceedingly power-stingy part, the PIC18F452 microcontroller can greatly extend battery life by alternating intervals of low-power sleep mode with intervals of normal operation. In terms of computation ability, PIC18F452 is capable of multiplication and division subroutines for multiple-byte, fixed-point numbers and for floating-point numbers.

Samples are available through Microchip and packaged in dual-inline packages so they are easy for prototyping on breadboards. Moreover, we have found and borrowed a reference guide for PIC18F452 called *Embedded Design with the PIC18F452 Microcontroller* [3], which will help us program PIC18F452. The book contains assembly code templates that encompasses some features of the microcontroller as well as its interactions with I/O pins.

For a highly accurate clock frequency and the significantly lower power dissipation that comes from a much lower clock frequency, for an accurate clock frequency in PIC18F452, the crystal provides great help. The output data from the colour sensor has a frequency range of 100kHz. We picked a 10 MHz to match a 2.5MHz clock rate in PIC18F452 and generated an overall 3MHz clock rate. This clock rate facilitates PIC18F452 to handle the colour data with no difficulty.

4.2.2. I/O Allocation

The circuit of the system is shown in the following schematic:

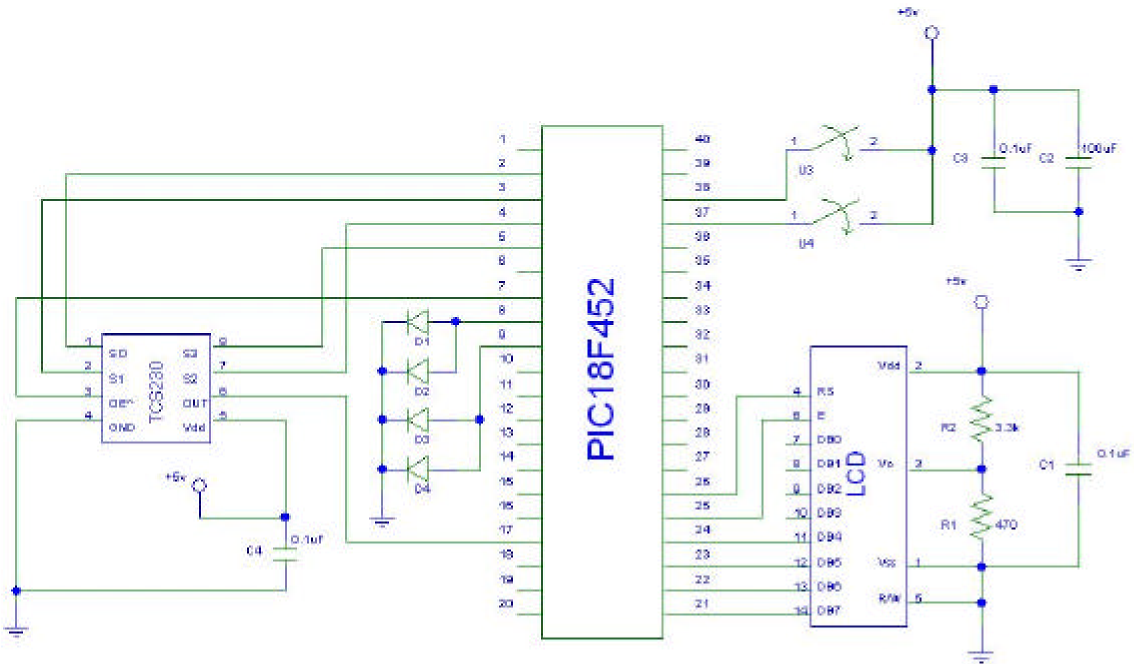


Figure 11: Overall Pin Assignment and Schematic of ColourIt

ColourIt uses battery power to sustain its operation. Power supplied to the system is up to +5V through a voltage regulator. D1, D2, D3, and D4 are white LEDs which are responsible for providing internal light source of ColourIt. U3 and U4 are power on switch and the toggle switch for LCD screen respectively. The capacitors are to sustain the voltage in case of a sudden current draw on the peripheral of the PIC18F452.

4.3. Control Software

4.3.1. Overview

The PIC18LF452 running Assembly with program code stored in Flash memory is the heart of the central control unit. The microcontroller interacts with the colour-sensing unit, processes data acquired from the colour sensors, display computation result externally on a LCD screen. Assembly firmware code is compiled in MPLAB IDE and simulated first with MPASMWIN which creates a virtual microcontroller environment in computer. Upon successful stimulation, the code will be loaded to microcontroller via the in-circuit-debugger, ICD2, where instruction execution is monitored in real-time.

We have designed a series of modules to efficiently compute input data from the colour sensor and present results in a user-friendly format.

4.3.2. White Calibration

The first step in using any color sensing system is to perform white calibration. When presented with a perfect white reflecting surface, an ideal RGB sensor would have equal signals on all three channels. However, due to differences in sensitivity of the red, green and blue channels, the RGB outputs are not equal. We can compensate for these differences by conducting experiments to determine how pure black and pure white appear on each channel. The test plan involves placing the sensor over pure black and pure white and recording the output clock cycles. Based on the results, outputs from the three channels are normalized through later calculations.

Choosing the right white reflecting surface is important since it is taken to be the maximum reflectance that each channel can detect. Our source of the white reflecting surface is obtained from a local professional photography shop, Lens and Shutters Cameras.

Hence, white calibration facilitates the pixel calculation module by providing the range of intensity difference in terms of clock cycles.

4.3.3. Main Control Module

Once the device is switched on, the microcontroller initializes the registers and memory then waits for a request from the user. The user has three options: Sample, Toggle content on LCD display, and Power Off. The following is the overview of the program flow.

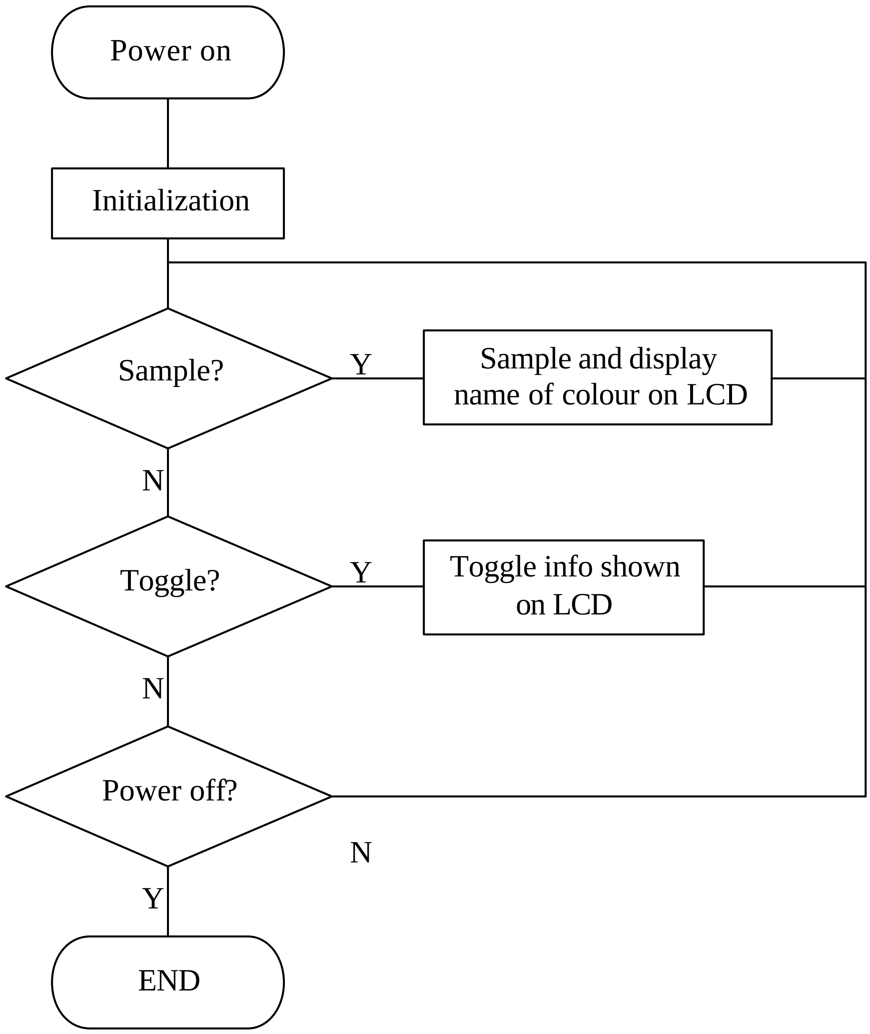


Figure 12: Overall Program Flow Chart

4.3.4. Controller Initialization Module

PIC18F452 allocates 1 byte of memory for every user defined variables in UDATA so they could be monitored real-time on the debugger, ICD2. Then, it assigns pointers to the main program, high priority, and low priority interrupts and non-volatile EEPROM for storage of colour-to-name lookup table.

4.3.5. Sample Intake Module

As a user requests a sample action, the microcontroller sends request for data to the colour-sensing module. Samples are obtained in the following order: clear, red, green, then blue channel. The output of the colour-sensing module is configured to be scaled down by 20% so we deal with smaller clock cycles in further calculation. Once the PIC18F452 receives data from the sensor, it utilizes the frequency measurement approach to determine intensity of each channel in terms of internal clock cycles. The number of clock cycles is in inverse proportion to the frequency of one colour channel. The more cycles we acquired through the algorithm, the lower the frequency of the colour channel. As a result there is less power in that colour channel.

4.3.6. Intensity Normalization and Pixel Calculation Module

With the clock cycle representations for pure black and pure white in each colour channel from previous experiments, we can perform white calibration and normalize intensity of each channel in spite of unequal photodiode sensitivities. White calibration is accomplished in the following sequence. First, compare the difference of clock cycles in each channel when the sensor is positioned over a white reflecting surface and a completely black surface. The clock cycles obtained from the sensor module are then taken into following calculation.

$$y = \frac{\text{max clock cycles - } x}{\text{max clock cycles} - \text{min clock cycles}}$$

After the calculation, outputs from red, green, blue channels are normalized. We then multiply y by full pixel scale, 255, to obtain an accurate pixel number in each channel.

4.3.7. Colour-to-Name Lookup Module

We plan to implement the colour-to-name lookup table in non-volatile memory such as the EEPROM. Writing to the data EEPROM is complicated since safeguards built in help ensure that only intended writes are carried out. The data stored in EEPROM is then secured from being overwritten. Colour names will be arranged in a stack with their pixel numbers as pointers. Stack will be divided into eight sections as follows. Each section will have at most twenty five entities.

Table 4: Major Sections in Colour-to-Name Lookup Table

	Red (0-127)	**Red (128-255)**
Green (0-127)	1	5
Green (128-255)	2	6
Blue (0-127)	3	7
Blue (128-255)	4	8

4.3.8. Output Display Module

After pixel numbers are obtained from intensity normalization and pixel calculation module, PIC18F452 reports the information externally to the user.

In our first stage prototype, we showed results in a 3 × 4 LED array. Each of the three columns representing red, green, and blue channels has four LEDs for indication of power level. Pixel values are divided equally into five levels: full scale of pixel value (in our case 255) lights all four LEDs. No LED will light up when a pixel number is less than 51.

For the complete system product, we will replace the LED arrays by a LCD screen with two lines and 8 characters on each line. We will utilize the DisplayC, DisplayV subroutines provided by *Embedded Design with the PIC18F452 microcontroller* for display of constant strings and variable values in RAM.

5. User Interface Unit

5.1. Overview

The interface unit is composed of an LCD module and two light touch switches (buttons). The LCD is used to display the information of the colour that is measured and analyzed by the device's colour sensor and microcontroller respectively. One of the buttons is used to activate the sampling function of the device for measuring and analyzing the colour of a desired spot. The other button is implemented to toggle different types of information that is displayed on the LCD when the button is pressed.

5.2. User Interface Hardware

5.2.1. User Interface Hardware

To display the information of the colour measured and analyzed by ColurIt to the user, we decide to integrate a light weight and economical LCD display with the system. The LCD used is made by Optrex Corporation, and its part number is DMC50448N-AAE- AD. We choose this LCD because of its low cost, and the simplicity of its interface with the microcontroller.

This LCD is capable of displaying two lines of characters. Each line can show as many as 8 characters, which is sufficient for our application. The minimum and maximum of the operating voltages are 4.5V and 5.5V respectively. The voltage supply of the system is 5V, so the LCD can be powered directly by the system's power supply without extra circuitry involved. Same goes with the connections between the LCD module and microcontroller. The LCD control output pins of the microcontroller can be connected directly to the LCD module.

The LCD module has 8 data bus pins for receiving data to be displayed from the microcontroller. As can be seen in Figure 13, only 4 data bus pins are connected to the microcontroller and it is because the LCD module can be programmed to operate under 4-pin operation mode. The complete pin assignment of the LCD module can be seen in Table 5: Pin Assignment of Optrex LCD:

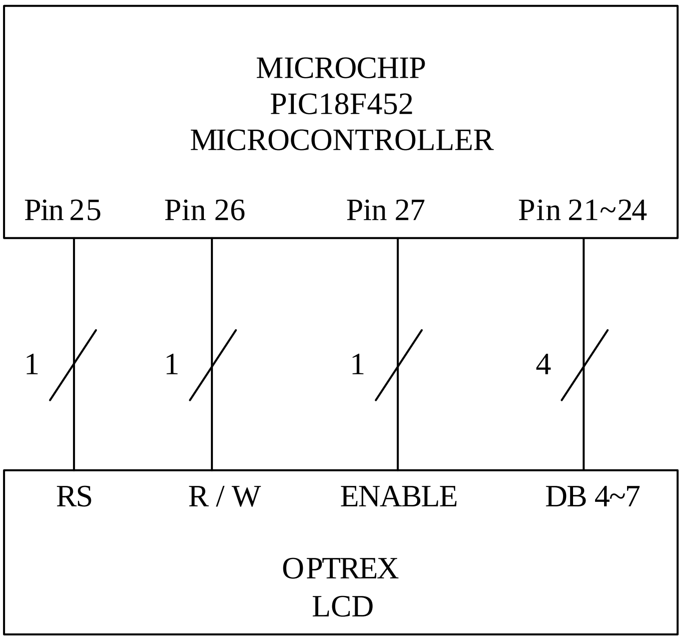


Figure 13: Block Diagram of Microcontroller and LCD

Table 5: Pin Assignment of Optrex LCD

No.	Symbol	Level	Function
1	V_{SS}	-	Power Supply (0V, GND)
2	V_{DD}	-	Power Supply for Logic
3	V_0	-	Power Supply for LCD Drive
4	RS	H / L	Register Select Signal
5	R / W	H / L	Read / Write Select Signal H : Read L : Write
6	E	H / L	Enable Signal (No pull-up Resister)
7	DB0	H / L	Data Bus Line / Non-connection at 4-bit operation
8	DB1	H / L	Data Bus Line / Non-connection at 4-bit operation
9	DB2	H / L	Data Bus Line / Non-connection at 4-bit operation
10	DB3	H / L	Data Bus Line /Non-connection at 4-bit operation
11	DB4	H / L	Data Bus Line
12	DB5	H / L	Data Bus Line
13	DB6	H / L	Data Bus Line
14	DB7	H / L	Data Bus Line

5.2.2. Buttons

The buttons of the system are implemented using the light touch switch EVQPAC04M made by Panasonic because of its low cost and simplicity. The type of the button is of top-push type. The circuit diagram is shown in Figure 14. When the button is pressed, A terminals will be shorted with B terminals. Upon release, the button returns to its original state.

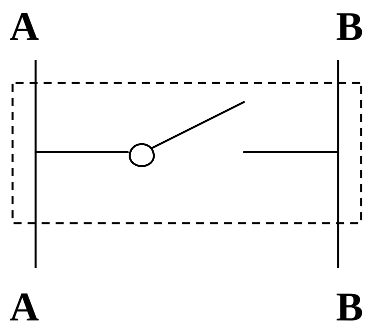


Figure 14: Circuit Diagram of Top-push Button

The button is connected to the microcontroller as shown in Figure 15. One terminal of the top-push button is connected to voltage high, and the other terminal is connected to an input pin of the microcontroller. According to Figure 15, pin 2 of the top-push button is connected to voltage low via a 1k ohm resistor. This allows the voltage level at pin 2 to be pulled down quickly after the button has been pressed and released. If pin 2 is not connected to ground via a resistor, the voltage level at pin 2 will drop very slowly after a button press, and it could potentially disturb the normal operation of the microprocessor.

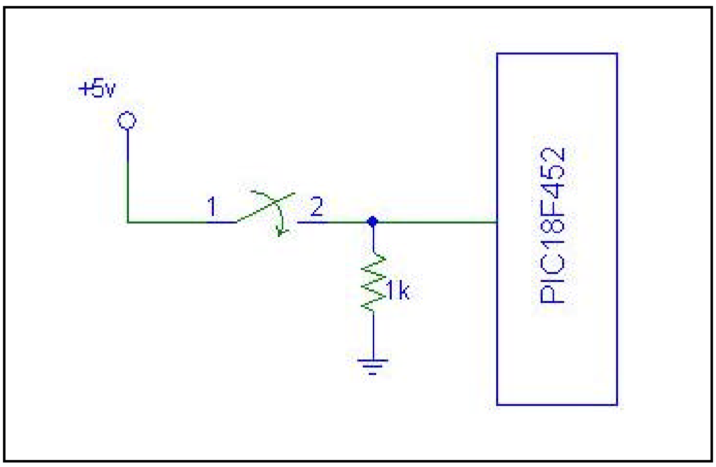


Figure 15: Circuit Connection of Top-push Button

Overall, three of this type of buttons will be used. The first button will activate the sampling function of the system. After the button is pressed, the microcontroller will gather data from the colour sensor, process the data, and output the information of the detected colour to the LCD display.

The second button will be used to toggle the information displayed on the LCD. Due to the LCD's limited capacity, only 16 characters can be displayed at the same time. After the sampling function of the system is finished, the pre-defined name of the detected colour will be shown on the LCD. The user can then press the button to toggle the LCD to show the pixel number of each RGB channel of the detected colour, and vice versa.

The third button is used to change the detection mode of the system. If the surface of an object to be detected is not a luminary, the system will need to illuminate the surface by turning on the 4 white LEDs placed beside the colour sensor. If the object is a luminary, then it is necessary that the 4 while LEDs not be turned on because it could potentially affect the accuracy of the result.

6. Packaging

6.1. Overview

Because ColourIt must be portable so that it is convenient to use in many applications, it must not weigh over 100g and can be no bigger than 4.40×2.00×12.00 cm to fit in most pockets and purses. The package must be able to contain and protect the internal circuitry, batteries and wiring. The LCD module should be positioned so that it can be easily viewed by the user. The power switch, sample and toggle button should be positioned so that it is easy for the user to control.

6.2. Container

Refer to the following diagram for the proposed design of the final packaging.

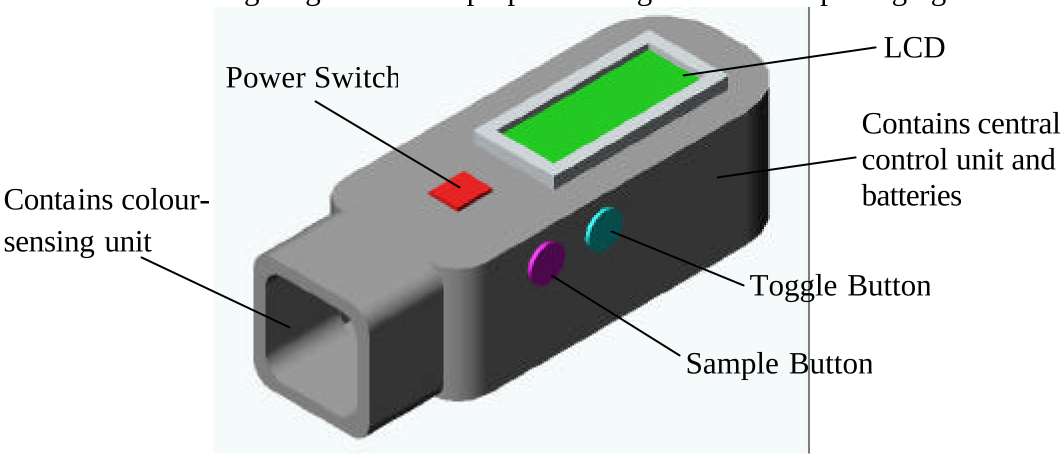


Figure 16: Integrated Packaging

As previously described (refer to Figure 9 and Figure 10), the colour-sensing unit must be able to hold the colour-sensor surrounded by four white LEDs. Therefore, the colour-sensing unit will be contained in a cuboid segment which is completely opaque.
The setup containing the control unit must also contain the batteries. The LCD module must be attached to this segment because it closely communicates with the microcontroller. The display is placed on the top so that it is easy to read for the user.
The sample button and toggle button are oriented so that the user can easily click the buttons with his/her right thumb. The two buttons are not too close together so that the user does not accidentally press the toggle button when trying to press the sample button.
The power switch is located on the top and apart from the other two buttons so that it is easily distinguished from the two function buttons.

7. Conclusion

The design specifications outlined in this document have described functions that apply to the entire system in many levels, which was determined through careful decisions based on feasibility, practicality and usability. The designs for component layout, implementation procedures, user interface specifications and packaging that must be incorporated for the successful development of Colo urIt. This imparts responsibility in both the designer and the user.

The engineers of ColourIt will follow this set of design specifications for the successful development of ColourIt.

8. Sources and References

[1] World Wide Web Consortium, "Recognized Colours and Keyword Names," http://www.w3.org/TR/SVG/types.html, January 2003

[2] Texas Advance Optoelectronic Solutions (TAOS), "Shaping the Future of Light Sensing Solutions," http://www.taosinc.com/downloads/pdf/brochure.pdf, 2005

[3] John B. Peatman, "Embedded Design with the PIC18F452 Microcontrolleres," Pearson Enducation, Inc., 2003

Appendix A: Colour Standards from World Wide Web Consortium

Table 6: 147 colours defined by world wide web consortium

aliceblue	rgb(240, 248, 255)	lightpink	rgb(255, 182, 193)
antiquewhite	rgb(250, 235, 215)	lightsalmon	rgb(255, 160, 122)
aqua	rgb(0, 255, 255)	lightseagreen	rgb(32, 178, 170)
aquamarine	rgb(127, 255, 212)	lightskyblue	rgb(135, 206, 250)
azure	rgb(240, 255, 255)	lightslategray	rgb(119, 136, 153)
beige	rgb(245, 245, 220)	lightslategrey	rgb(119, 136, 153)
bisque	rgb(255, 228, 196)	lightsteelblue	rgb(176, 196, 222)
black	rgb(0, 0, 0)	lightyellow	rgb(255, 255, 224)
blanchedalmond	rgb(255, 235, 205)	lime	rgb(0, 255, 0)
blue	rgb(0, 0, 255)	limegreen	rgb(50, 205, 50)
blueviolet	rgb(138, 43, 226)	linen	rgb(250, 240, 230)
brown	rgb(165, 42, 42)	magenta	rgb(255, 0, 255)
burlywood	rgb(222, 184, 135)	maroon	rgb(128, 0, 0)
cadetblue	rgb(95, 158, 160)	mediumaquamarine	rgb(102, 205, 170)
chartreuse	rgb(127, 255, 0)	mediumblue	rgb(0, 0, 205)

Table 7: 147 colours defined by world wide web consortium (Cont' d)

	coral	rgb(255, 127, 80)		mediumpurple	rgb(147, 112, 219)
	cornflowerblue	rgb(100, 149, 237)		mediumseagreen	rgb(60, 179, 113)
	cornsilk	rgb(255, 248, 220)		mediumslateblue	rgb(123, 104, 238)
	crimson	rgb(220, 20, 60)		mediumspringgreen	rgb(0, 250, 154)
	cyan	rgb(0, 255, 255)		mediumturquoise	rgb(72, 209, 204)
	darkblue	rgb(0, 0, 139)		mediumvioletred	rgb(199, 21, 133)
	darkcyan	rgb(0, 139, 139)		midnightblue	rgb(25, 25, 112)
	darkgoldenrod	rgb(184, 134, 11)		mintcream	rgb(245, 255, 250)
	darkgray	rgb(169, 169, 169)		mistyrose	rgb(255, 228, 225)
	darkgreen	rgb(0, 100, 0)		moccasin	rgb(255, 228, 181)
	darkgrey	rgb(169, 169, 169)		navajowhite	rgb(255, 222, 173)
	darkkhaki	rgb(189, 183, 107)		navy	rgb(0, 0, 128)
	darkmagenta	rgb(139, 0, 139)		oldlace	rgb(253, 245, 230)
	darkolivegreen	rgb(85, 107, 47)		olive	rgb(128, 128, 0)
	darkorange	rgb(255, 140, 0)		olivedrab	rgb(107, 142, 35)
	darkorchid	rgb(153, 50, 204)		orange	rgb(255, 165, 0)
	darkred	rgb(139, 0, 0)		orangered	rgb(255, 69, 0)

Table 8: 147 colours defined by world wide web consortium (Cont' d)

	darkseagreen	rgb(143, 188, 143)		palegoldenrod	rgb(238, 232, 170)
	darkslateblue	rgb(72, 61, 139)		palegreen	rgb(152, 251, 152)
	darkslategray	rgb(47, 79, 79)		paleturquois e	rgb(175, 238, 238)
	darkslategrey	rgb(47, 79, 79)		palevioletred	rgb(219, 112, 147)
	darkturquoise	rgb(0, 206, 209)		papayawhip	rgb(255, 239, 213)
	darkviolet	rgb(148, 0, 211)		peachpuff	rgb(255, 218, 185)
	deeppink	rgb(255, 20, 147)		peru	rgb(205, 133, 63)
	deepskyblue	rgb(0, 191, 255)		pink	rgb(255, 192, 203)
	dimgray	rgb(105, 105, 105)		plum	rgb(221, 160, 221)
	dimgrey	rgb(105, 105, 105)		powderblue	rgb(176, 224, 230)
	dodgerblue	rgb(30, 144, 255)		purple	rgb(128, 0, 128)
	firebrick	rgb(178, 34, 34)		red	rgb(255, 0, 0)
	floralwhite	rgb(255, 250, 240)		rosybrown	rgb(188, 143, 143)
	forestgreen	rgb(34, 139, 34)		royalblue	rgb(65, 105, 225)
	fuchsia	rgb(255, 0, 255)		saddlebrown	rgb(139, 69, 19)
	gainsboro	rgb(220, 220, 220)		salmon	rgb(250, 128, 114)
	ghostwhite	rgb(248, 248, 255)		sandybrown	rgb(244, 164, 96)

Table 9: 147 colours defined by world wide web consortium (Cont' d)

	goldenrod	rgb(218, 165, 32)		seashell	rgb(255, 245, 238)
	gray	rgb(128, 128, 128)		sienna	rgb(160, 82, 45)
	grey	rgb(128, 128, 128)		silver	rgb(192, 192, 192)
	green	rgb(0, 128, 0)		skyblue	rgb(135, 206, 235)
	greenyellow	rgb(173, 255, 47)		slateblue	rgb(106, 90, 205)
	honeydew	rgb(240, 255, 240)		slategray	rgb(112, 128, 144)
	hotpink	rgb(255, 105, 180)		slategrey	rgb(112, 128, 144)
	indianred	rgb(205, 92, 92)		snow	rgb(255, 250, 250)
	indigo	rgb(75, 0, 130)		springgreen	rgb(0, 255, 127)
	ivory	rgb(255, 255, 240)		steelblue	rgb(70, 130, 180)
	khaki	rgb(240, 230, 140)		tan	rgb(210, 180, 140)
	lavender	rgb(230, 230, 250)		teal	rgb(0, 128, 128)
	lavenderblush	rgb(255, 240, 245)		thistle	rgb(216, 191, 216)
	lawngreen	rgb(124, 252, 0)		tomato	rgb(255, 99, 71)
	lemonchiffon	rgb(255, 250, 205)		turquoise	rgb(64, 224, 208)
	lightblue	rgb(173, 216, 230)		violet	rgb(238, 130, 238)
	lightcoral	rgb(240, 128, 128)		wheat	rgb(245, 222, 179)

Table 10: 147 colours defined by world wide web consortium (Cont' d)

	lightcyan	rgb(224, 255, 255)		white	rgb(255, 255, 255)
	lightgoldenrodyellow	rgb(250, 250, 210)		whitesmoke	rgb(245, 245, 245)
	lightgray	rgb(211, 211, 211)		yellow	rgb(255, 255, 0)
	lightgreen	rgb(144, 238, 144)		yellowgreen	rgb(154, 205, 50)
	lightgrey	rgb(211, 211, 211)			

Appendix B: PIC18XXX

Table 11: Timer modules built in PIC18FXXX

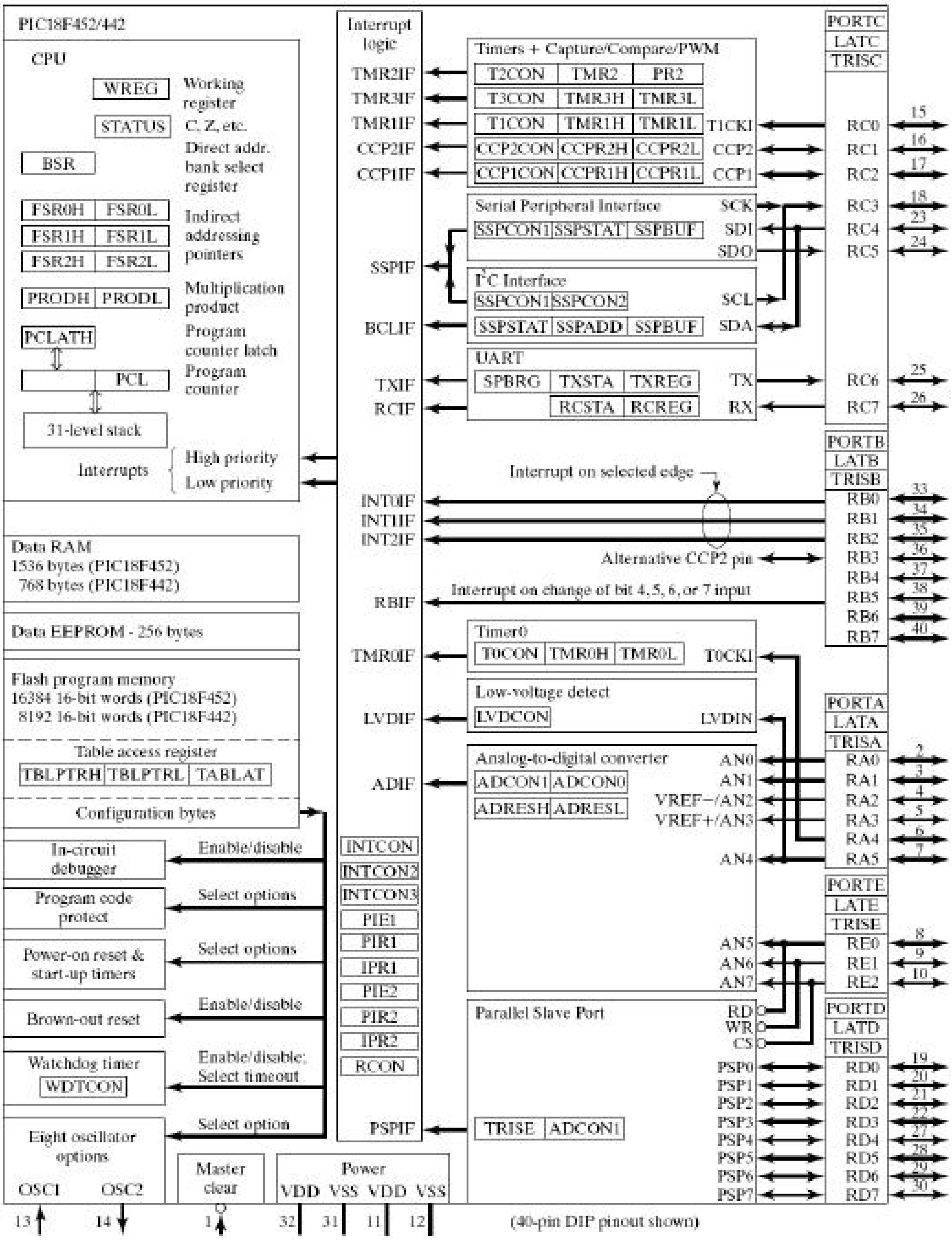
Timer	Function
Timer0	8-bit/16-bit timer/counter with 8-bit programmable prescaler
Timer1	16-bit timer/counter
Timer2	8-bit timer/counter with 8-bit period register (timer-base for PWM)
Timer3	16-bit timer/counter

Table 12: Alternative family member parts

Part Number	Program Memory (16-bit words)	RAM Bytes	Total Pins	I/O Pins	Package size—length × width, including pins				
					40-pin DIP	44-pin PLCC	44-pin TQFP	28-pin DIP	28-pin SOIC
PIC18F452	16384	1536	40/44	33	2.058″ × 0.600″	0.690″ × 0.690″	0.472″ × 0.472″		
PIC18F442	8192	768	40/44	33	2.058″ × 0.600″	0.690″ × 0.690″	0.472″ × 0.472″		
PIC18F252	16384	1536	28	22				1.345″ × 0.300″	0.704″ × 0.407″
PIC18F242	8192	768	28	22				1.345″ × 0.300″	0.704″ × 0.407″

Appendix C: Port, Pin and Register Allocation of PIC18F452

Table 13: Map of Port, Pin and Register in PIC18F452



Appendix D: Instruction Set of PIC18F452

Table 14: Instruction sets of PIC18F452

Mnemonic	Operands	Description	Words	Cycles	Status bits affected
movlw	k	Move literal value to WREG	1	1	-
movwf	f(, BANKED)	Move WREG to f	1	1	-
movff	f_S, f_D	Move f_S to f_D (both with full 12-bit addresses)	2	2	-
movf	f, F/W(, BANKED)	Move f to F or WREG	1	1	Z, N
lfsr	i, k	Load FSRi with full 12-bit address, where i = 0 to 2	2	2	-
clrf	f(, BANKED)	Clear f	1	1	Z
setf	f(, BANKED)	0xff → f	1	1	-
movlb	k	Move literal value to BSR<3 : 0> , where k = 0 to 15, to set the bank for direct addressing	1	1	-
swapf	f, F/W(, BANKED)	Swap nibbles of f, putting result in F or WREG	1	1	-
bcf	f, b(, BANKED)	Clear bit b of register f, where b = 0 to 7	1	1	-
bsf	f, b(, BANKED)	Set bit b of register f, where b = 0 to 7	1	1	-
btg	f, b(, BANKED)	Toggle bit b of register f, where b = 0 to 7	1	1	-
rlcf	f, F/W(, BANKED)	Copy f into F or WREG; rotate F or WREG left through carry bit (9-bit rotate left)	1	1	C, N, Z
rlncf	f, F/W(, BANKED)	Copy f into F or WREG; rotate F or WREG left without carry bit (8-bit rotate left)	1	1	N, Z
rrcf	f, F/W(, BANKED)	Copy f into F or WREG; rotate F or WREG right through carry bit (9-bit rotate right)	1	1	C, N, Z
rrncf	f, F/W(, BANKED)	Copy f into F or WREG; rotate F or WREG right without carry bit (8-bit rotate right)	1	1	N, Z
incf	f, F/W(, BANKED)	Increment f, putting result in F or WREG	1	1	C, DC, Z, OV, N
decf	f, F/W(, BANKED)	Decrement f, putting result in F or WREG	1	1	C, DC, Z, OV, N
comf	f, F/W(, BANKED)	Complement f, putting result in F or WREG	1	1	Z, N
negf	f(, BANKED)	Change sign of a twos-complement-coded number	1	1	C, DC, Z, OV, N
andlw	k	AND literal value into WREG	1	1	Z, N
andwf	f, F/W(, BANKED)	AND WREG with f, putting result in F or WREG	1	1	Z, N
iorlw	k	Inclusive-OR literal value into WREG	1	1	Z, N
iorwf	f, F/W(, BANKED)	Inclusive-OR WREG with f, putting result in F or WREG	1	1	Z, N
xorlw	k	Exclusive-OR literal value into WREG	1	1	Z, N
xorwf	f, F/W(, BANKED)	Exclusive-OR WREG with f, putting result in F or WREG	1	1	Z, N
addlw	k	Add literal value into WREG	1	1	C, DC, Z, OV, N
addwf	f, F/W(, BANKED)	Add WREG and f, putting result in F or WREG	1	1	C, DC, Z, OV, N
addwfc	f, F/W(, BANKED)	Add WREG and f and carry bit, putting result in F or WREG	1	1	C, DC, Z, OV, N
daw		Decimal adjust sum of two packed BCD digits to correct packed BCD result in WREG	1	1	C
sublw	k	Subtract WREG from literal value, putting result in WREG	1	1	C, DC, Z, OV, N
subwf	f, F/W(, BANKED)	Subtract WREG from f, putting result in f or WREG	1	1	C, DC, Z, OV, N
subwfb	f, F/W(, BANKED)	Subtract WREG and borrow bit from f, putting result in F or WREG	1	1	C, DC, Z, OV, N
subfwb	f, F/W(, BANKED)	Subtract f and borrow bit from WREG, putting result in F or WREG	1	1	C, DC, Z, OV, N
mullw	k	Multiply WREG with literal value, putting result in PRODH : PRODL (WREG remains unchanged)	1	1	-
mulwf	f(, BANKED)	Multiply WREG with f, putting result in PRODH : PRODL (WREG and f remain unchanged)	1	1	-
btfsc	f, b(, BANKED)	Test bit b of register f, where b = 0 to 7 ; skip if clear	1	1/2	-
btfss	f, b(, BANKED)	Test bit b of register f, where b = 0 to 7; skip if set	1	1/2	-

Table 15: Instruction sets of PIC18F452 (Cont' d)

Mnemonic	Operands	Description	Words	Cycles	Status bits affected
bra	label	Branch to labeled instruction (within ±64 one-word instructions)	1	2	-
goto	label	Go to labeled instruction (anywhere)	2	2	-
bc	label	If carry (C=1), then branch to labeled instruction (within ±64 one-word instructions)	1	1/2	-
bnc	label	If no carry (C=0), then branch to labeled instruction (within ±64 one-word instructions)	1	1/2	-
bz	label	If zero (Z=1), then branch to labeled intruction (within ±64 one-word instructions)	1	1/2	-
bnz	label	If not zero (Z=0, then branch to labeled instruction (within ±64 one-word instructions)	1	1/2	-
bn	label	If negative (N=1), then branch to labeled instruction (within ±64 one-word instructions)	1	1/2	-
bnn	label	If not negative (N=0), then branch to labeled instruction (within ±64 one-word instructions)	1	1/2	-
bov	label	If overflow (OV=1), then branch to labeled instruction (within ±64 one-word instructions)	1	1/2	-
bnov	label	If no overflow (OV=0), then branch to labeled instruction (within ±64 one-word instructions)	1	1/2	-
cpfseq	f(, BANKED)	Skip if f is equal to WREG	1	1/2	-
cpfsgt	f(, BANKED)	Skip if f is greater than WREG (unsigned compare)	1	1/2	-
cpfslt	f(, BANKED)	Skip if f is less than WREG (unsigned compare)	1	1/2	-
tstfsz	f(, BANKED)	Test f; skip if zero	1	1/2	-
decfsz	f, F/W(, BANKED)	Decrement f, putting result in F or WREG; skip if zero	1	1/2	-
dcfsnz	f, F/W(, BANKED)	Decrement f, putting result in F or WREG; skip if not zero	1	1/2	-
incfsz	f, F/W(, BANKED)	Increment f, putting result in F or WREG; skip if zero	1	1/2	-
infsnz	f, F/W(, BANKED)	Increment f, putting result in F or WREG; skip if not zero	1	1/2	-
rcall	label	Call labeled subroutine (within ±512 one-word instructions)	1	2	-
call	label	Call labeled subroutine (anywhere)	2	2	-
call	label, FAST	Call labeled subroutine (anywhere); copy state to shadow registers: (WREG)→WS, (STATUS)→STATUSS, (BSR)→BSRS	2	2	-
return		Return from subroutine	1	2	-
return	FAST	Return from subroutine; restore state from shadow registers: (WS)→WREG, (STATUSS)→STATUS, (BSRS)→BSR	1	2	C, DC, Z, OV, N
retlw	k	Return from subroutine, putting literal value in WREG	1	2	-
retfie		Return from interrupt; reenable interrupts	1	2	-
retfie	FAST	Return from interrupt; restore state from shadow registers: (WS)→WREG, (STATUSS)→STATUS, (BSRS)→BSR; reenable interrupts	1	2	C, DC, Z, OV, N
push		Push address of next instruction onto stack	1	1	-
pop		Discard address on top of stack	1	1	-
clrwdt		Clear watchdog timer	1	1	-
sleep		Go into standby mode	1	1	-
reset		Software reset to same state as is achieved with the MCLR input	1	1	C, DC, Z, OV, N
nop		No operation	1	1	-
tblrd*		Read from program memory location pointed to by TBLPTR into TABLAT	1	2	-
tblrd*+		Read from program memory location pointed to by TBLPTR into TABLAT, then increment TBLPTR	1	2	-
tblrd*-		Read from program memory location pointed to by TBLPTR into TABLAT, then decrement TBLPTR	1	2	-
tblrd+*		Increment TBLPTR, then read from program memory location pointed to by TBLPTR into TABLAT	1	2	-