AC 2010-1527: PERSONAL LAB HARDWARE: A SINE WAVE GENERATOR, LOGIC PULSE SIGNAL, AND PROGRAMMABLE SYNCHRONOUS SERIAL INTERFACE FOR ENHANCING EDUCATION

Jeffrey Richardson, Purdue University

James Jacob, Purdue University

Brant Price, Purdue University

Jeremiah Dole, Purdue University

Personal Lab Hardware: A Sine Wave Generator, Logic Pulse Signal, and Programmable Synchronous Serial Interface for Enhancing Education

Abstract:

This project developed a low cost solution to provide personally owned electronic test equipment to students in an advanced analog electronics course to enhance their learning associated with electronic laboratory experiments. The project included the development of a sine wave generator, logic pulse signal, and a programmable synchronous serial interface. The project is based a Cypress programmable system on a chip or PSoC and a LabVIEW PC application. The PSoC allows for a single chip solution of the sine wave generator by creating a programmable square wave that is then sent through an internal 4-pole filter to shape the desired sine wave. This approach is novel when compared to direct digital synthesis or look-up tables. The PSoC also creates a programmable logic pulse output to provide a variable duty cycle square when required. The programmable synchronous serial interface allows students to select the appropriate timing diagram and then specify the desired data to be transferred. The desired data is then transfer to the PSoC which ultimately controls the corresponding synchronous serial device eliminating the need for the analog student to write software to control the various devices. This feature allows students to study synchronous serial devices such as DAC's, ADC's, digital potentiometers, etc. without the confusion sometimes associated with programming a microcontroller to accomplish the task. This allows the students to concentrate on the analog electronics at hand, and not waste time attempting to create software. The project allows the students to focus their time and energy on the analog fundamentals and ultimately enhances their educational experiences. The overall design of the project, the educational benefits, and assessment data from the use of the project will be presented in this paper.

Introduction:

Many engineering and technology programs rely on hands-on application of the topics studied in the classroom. During a normally scheduled laboratory, the electrical engineering and technology students demonstrate the operation of an electric circuit to their laboratory instructor. In an effort to better prepare the students for their laboratory session and reduce the amount of time the students and laboratory instructors spend in the formal laboratory environment, a system of personally owned, student hardware was created to allow students to perform any necessary work at the student's individual residence at a low cost. This allows the students to spend as much or as little time on a topic as is required for mastery prior to attending their laboratory session. Competency is then demonstrated to the lab instructor in the department's laboratories using professional-grade equipment.

The original project included the creation of a custom LabVIEW application and a custom circuit to provide a function generator and a two channel oscilloscope. To increase the capability and overall usefulness of the system, a programmable logic compatible square wave was created with a frequency range from 10Hz to 100kHz with an with adjustable duty cycle from 0 to 100%. In addition to the square wave generator, a programmable synchronous serial interface was also created. This feature allows students to study synchronous serial devices such as digital to analog converters (DAC's), analog to digital converters (ADC's), direct digital synthesis (DDS), and digital potentiometers, etc. without the confusion sometimes associated with programming a microcontroller to accomplish the task. The programmable square wave generator and programmable synchronous serial interface along with the LabVIEW interface are the focus of this paper.

Educational Need:

Electrical and Computer Engineering Technology 36800 at Purdue University is an advanced analog electronics course that focuses on linear integrated circuits. The students in this course study of the applications of analog integrated circuits through topics like linear amplifiers, IC specifications, linear regulators, waveform generation, linear and switched capacitor active filters, and nonlinear circuit applications. As part of this course, it is desired to utilize a multitude of devices which require a synchronous serial interface typically generated by a microcontroller. However, as noted earlier, this is an advanced analog course and the students enrolled in the course historically have struggled to implement the necessary software to implement the synchronous serial interfaces required to apply these devices in their circuits. The instructors of this course therefore desired a solution to overcome this short coming [1].

Education Solution:

As a solution to this challenge, the instructors have turned towards a custom LabVIEW interface coupled with a custom electronic design. During a previous project, a custom circuit was created to provide the students with a two channel digital storage type oscilloscope and sine wave generator. This system is based around a Cypress CY8C29466 Programmable System on a Chip or PSoC from Cypress Semiconductor. A PSoC is a very powerful and flexible microcontroller system. The flexible nature of the PSoC allowed the additional functionality to be added to the existing system by simply adding a few connectors to the hardware, adding the required software to the PSoC, and the creation of required LabVIEW interface. A picture of the hardware prototype for the system is shown below in figure 1.

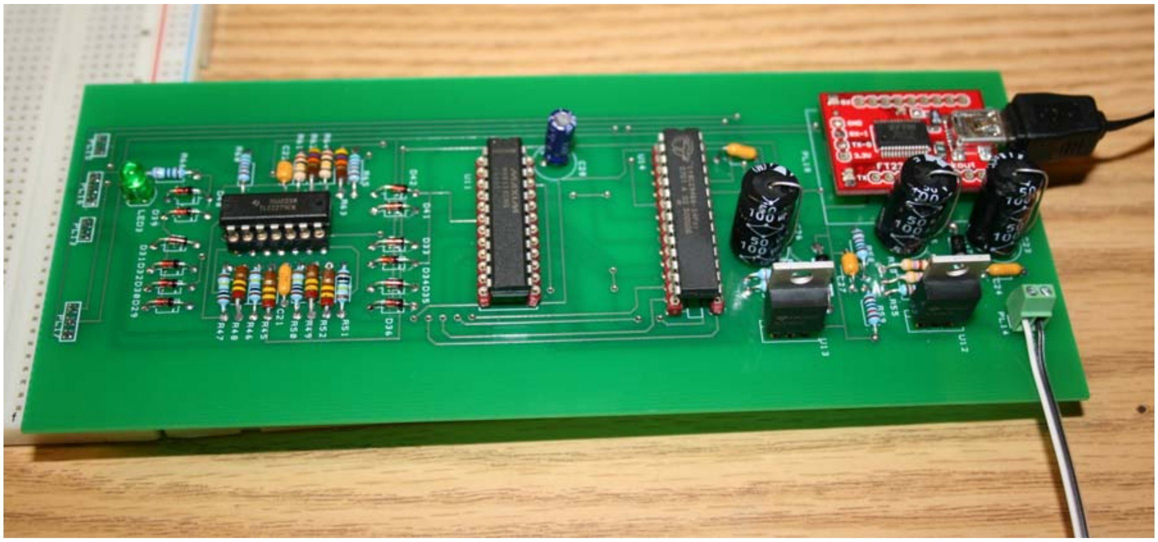


Figure 1 – Personal Laboratory Hardware Prototype

Logic Output:
The first new feature added to the design was a programmable logic pulse generator capable of providing a square wave with an adjustable duty cycle from 10Hz to 100kHz over a duty cycle range of 0% to 100%. In order to create the desired waveform, a 24-bit counter circuit was implemented inside the PSoC. The 24-bit counter is shown below in figure 2. The 24-bit counter allows the range required to provide the wide output range. In order to implement a counter, two values are required: a period value to control the frequency and a compare value to control the duty cycle. The period value is calculated by dividing the system clock by the desired frequency and the compare value is calculated by multiplying the calculated period value by the desired duty cycle divided by 100 as seen in figure 3 and 4.

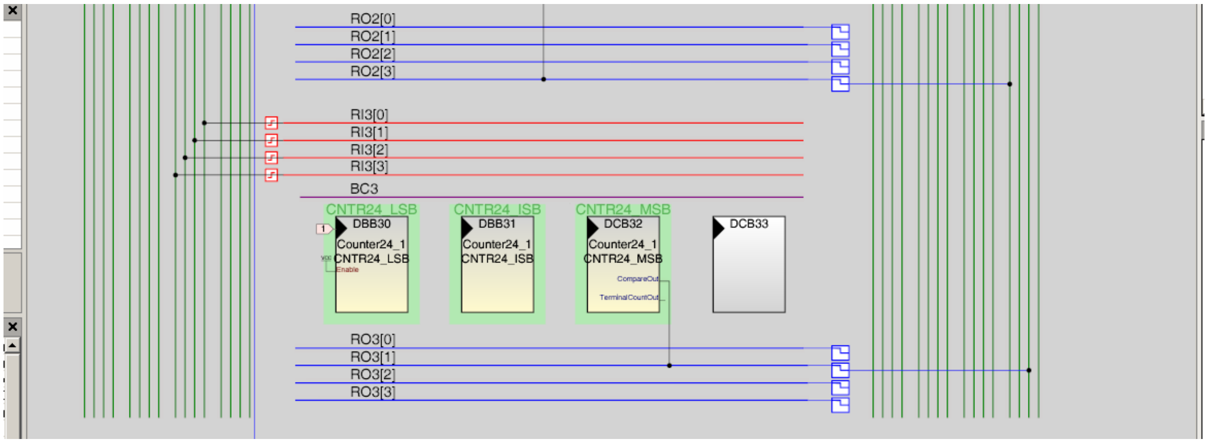


Figure 2 – 24-Bit Counter in the PSoC

$$\text{period value} = \frac{\text{24MHz}}{\text{desired frequency}}$$

Figure 3 – Equation to Calculate Period for Frequency Control

$$\text{compare value} = \frac{(\text{period value* desired duty cycle})}{\text{period value}}$$

Figure 4 – Compare Value Calculation for Duty Cycle Control

The output of the counter circuit is sent through an operational amplifier circuit configured as a buffer. The output of the operational amplifier is connected to a set of 1N914 small signal diodes to clamp the output in the event of an over-voltage situation caused by a wiring error by a student user. The schematic diagram of the output protection circuit is shown in figure 5.

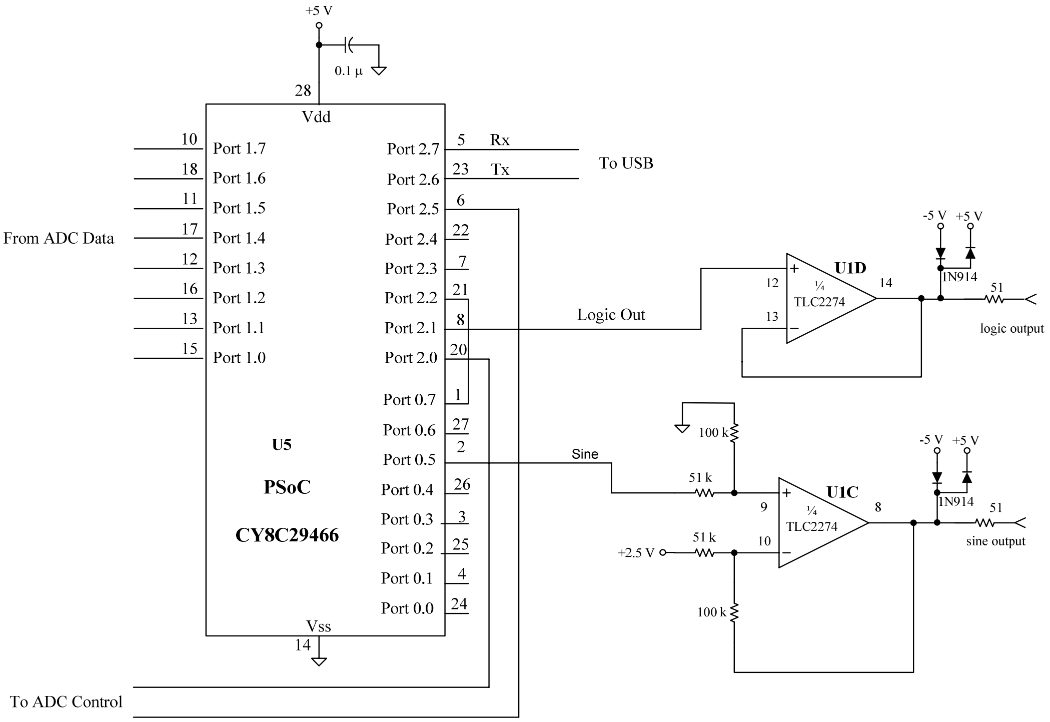


Figure 5 – Logic Output Schematic

The software to create the pulse output is a straight forward implementation of the equations shown on the previous page in figures 3 & 4. The LabVIEW interface sends a single byte control character followed by the desired frequency and desired duty cycle. These values are then used to calculate the required values which are then loaded into the appropriate control registers as shown below in figure 6.

```
// get a value from LabVIEW for the desired frequency
counter_frequency = get_frequency();
counter_value = 24000000 / counter_frequency;
// get a value from the compare value of the counter to control the Duty Cycle
DutyCycle_value = get_dutycycle();
compare_value = (counter_value * DutyCycle_value) / 100;


Counter24_1_Start();
Counter24_1_WritePeriod(  counter_value );
Counter24_1_WriteCompareValue( compare_value );
```

Figure 6 – Software Code Example

The overall software flowchart for the PSoC is shown below in figure 7. The PSoC monitors a serial connection for control data bytes. Once a control byte has been received, the software makes an appropriate decision and then either waits for additional information before carrying out the desired operation, or in some instances, the PSoC only requires a single control byte before performing the desired action and then sending data back to the LabVIEW control application.

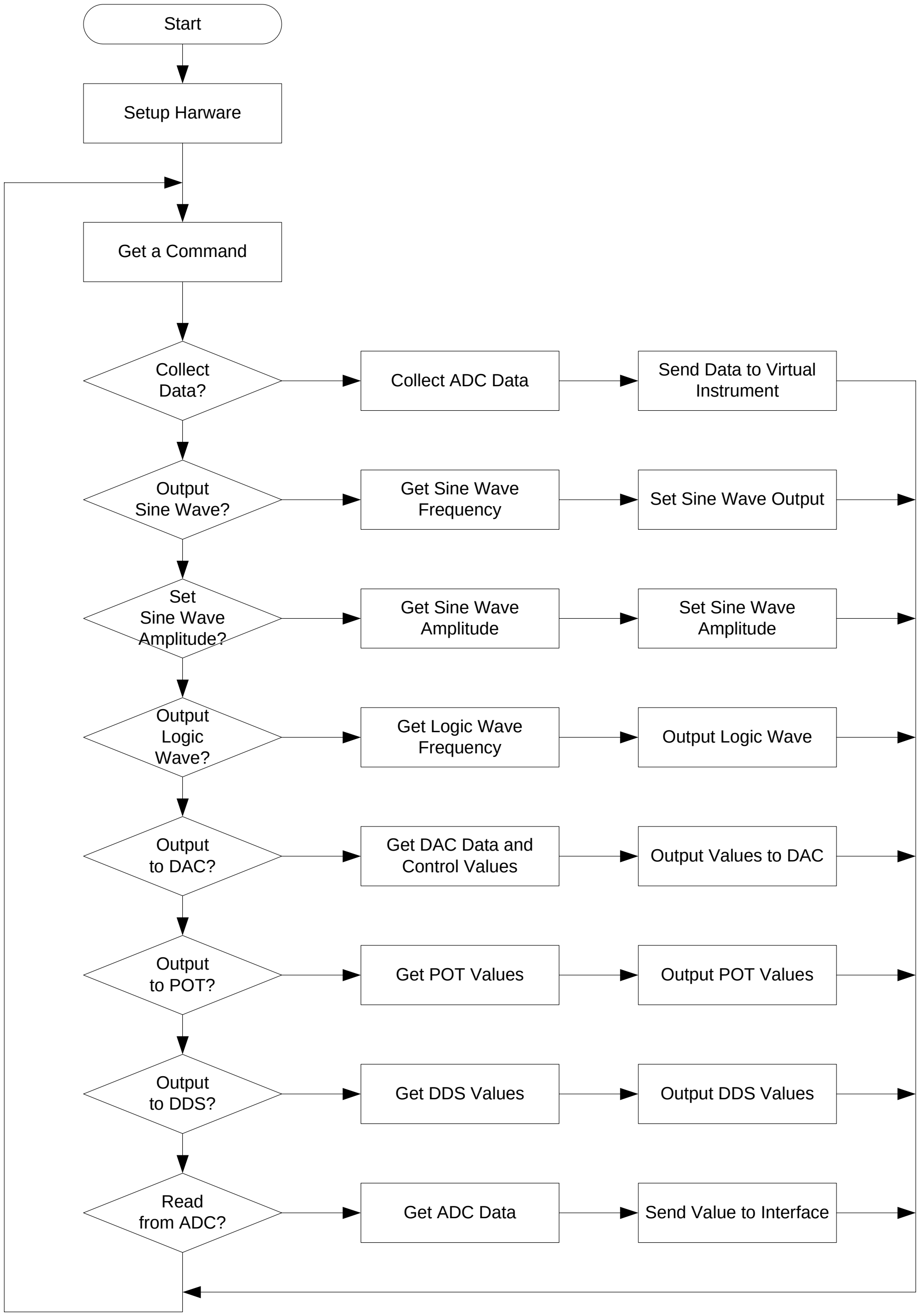


Figure 7 – Overall Software Flowchart

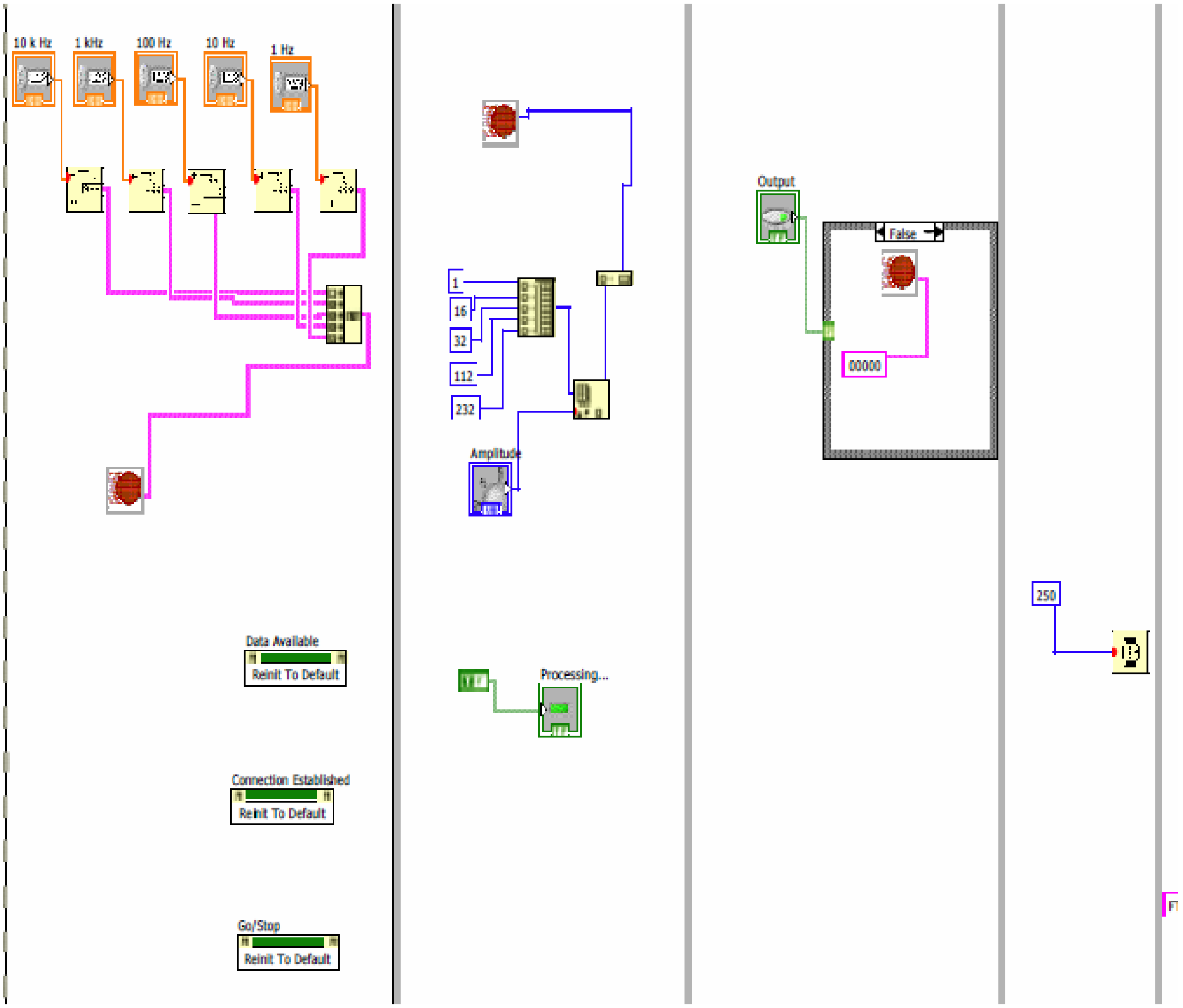
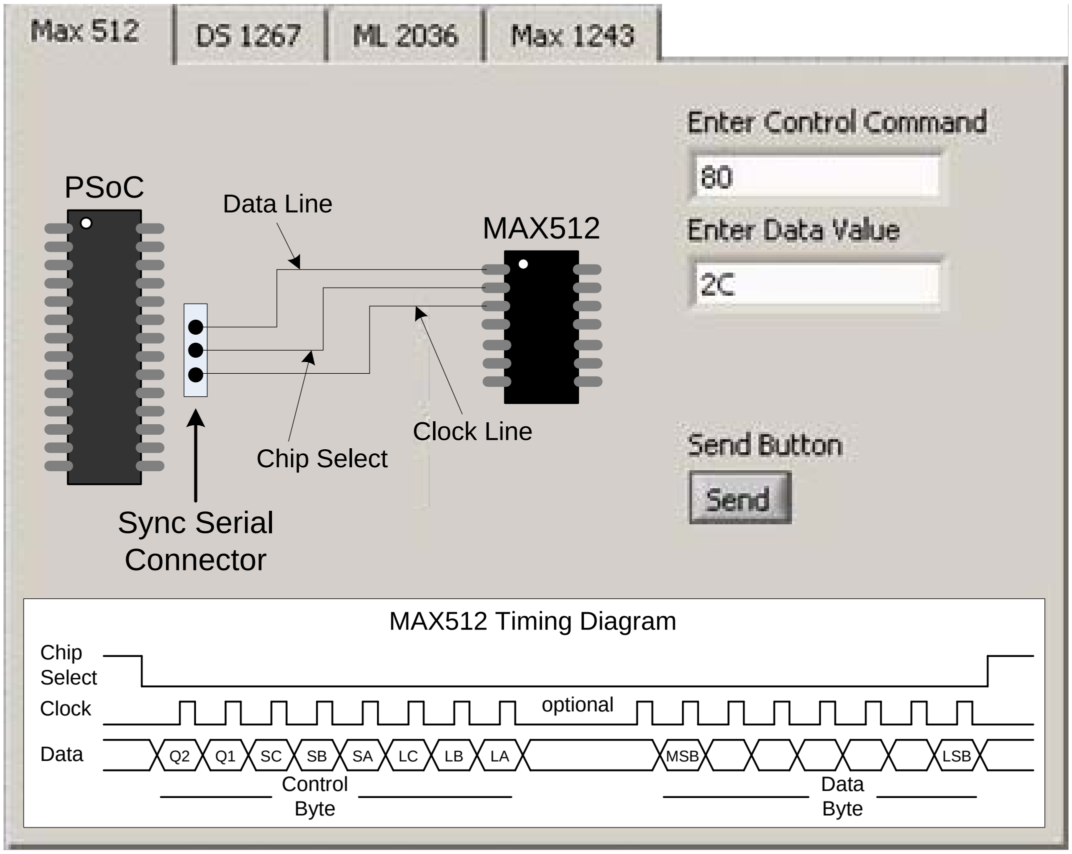


Figure 8 – LabVIEW Communication and Control

The LabVIEW application contains multiple sequences to control and communication with the PSoC. The first sequence shown in figure 8 above is typical for the control over the frequency of the logic output. The value for each of the digits is set and then sent to the PSoC.

Synchronous Serial Device Control:

Perhaps the most beneficial addition this project adds to the laboratory experience is the ability to communicate with a multitude of synchronous serial devices without the need for the student to actually write software. This feature allows the advanced analog students to concentrate on the application of the devices and not the subtle details associated with the timing diagrams for the devices. However, the students are still responsible to determining the values that must be written to the devices. The LabVIEW interface provides a mechanism to transfer data, but the student is still responsible for making sure the data is correct. A typical synchronous serial interface control is shown in figure 9 on the next page.



Enter Control Command: 80

Enter Data Value: 2C

Figure 9 – Typical Synchronous Serial Interface

The interface is comprised of multiple tabs that allow the student to select a specific device type. The individual timing diagrams for each of these devices is shown in figure 10 to illustrate the differences between them.

The PSoC software required to transfer data between the LabVIEW application and the physical synchronous serial device is broken into two main components. A decision is made in the main loop of the PSoC software to determine that the user desires to update the value in one of the synchronous serial devices. The LabVIEW application then sends the additional control and data values to the PSoC. Depending on which device is being updated, a specific function is executed to actually transfer the data from the PSoC to the target synchronous serial device. The main software that receives the command and control data from LabVIEW is shown in figure 11. The actual software function to handles the transfer of the data to the target device is shown in figure 12.

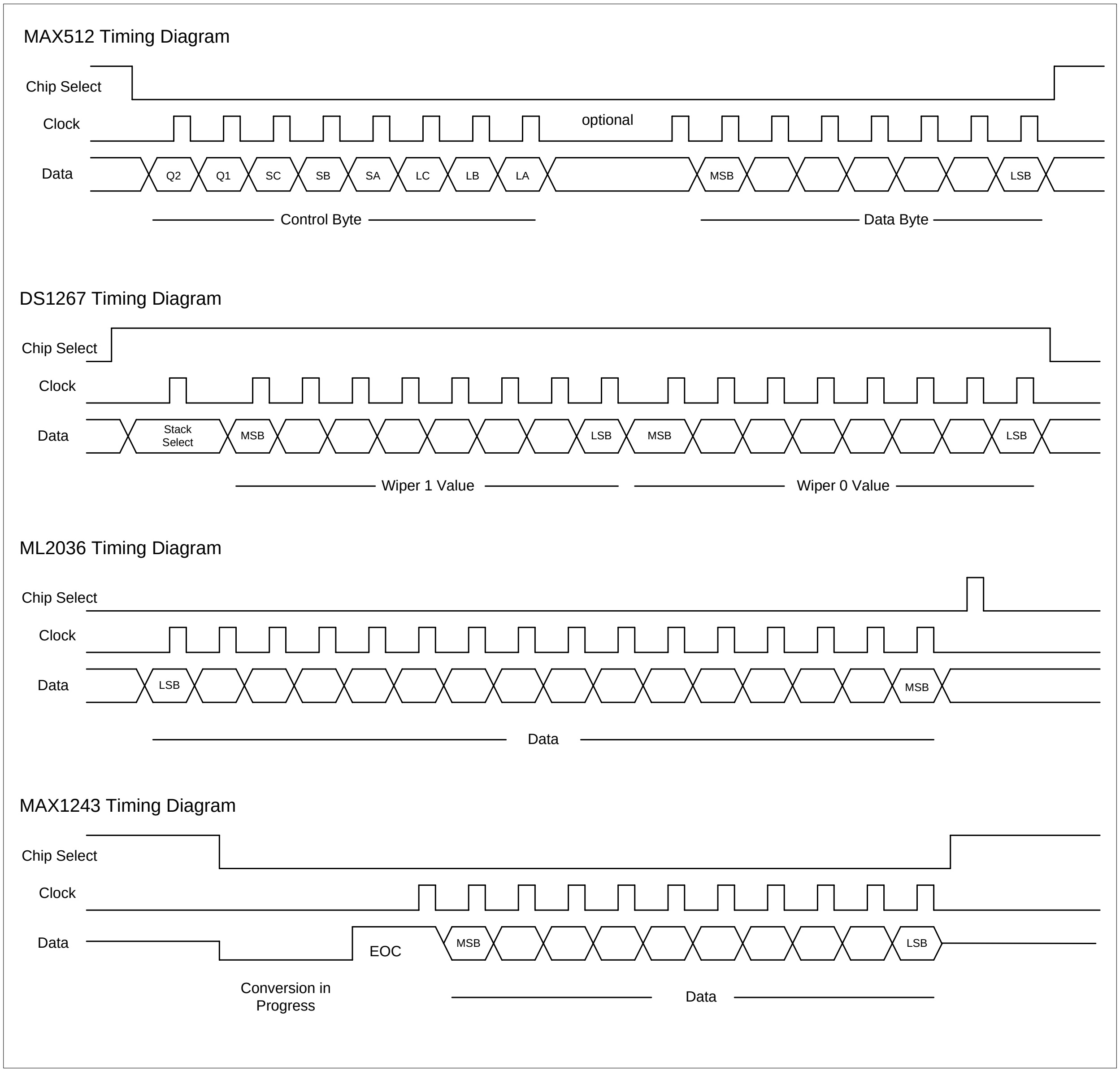


Figure 10 – Timing Diagrams of Individual Devices

```
case 0x19:                  // write data to the MAX512
          control = get_2Digit_Hex();      // get the 8-bit control value from LabVIEW
          value = get_2Digit_Hex();        // get the 8-bit data value for the DAC
          send_MAX512(control, value);  // send the values to the target device
          break;
```

Figure 11 – Main Software for Data Transfer

```
void send_MAX512(unsigned char control, unsigned char dac_data)
        {
        unsigned char mask;
        unsigned char i;


        CLR_CS;

        mask = 0x80;                                    // setup mask for MSB first
        for(i = 0; i < 8; i++)                  // send the address bits first
                {
                if((control & mask) != 0)       // this bit is set???
                        {
                        SET_DATA;                               // set the data pin high
                        }
                else
                        {
                        CLR_DATA;                               // or set it low.
                        }
                SET_CLK;                                        // work the clock...
                mask = mask >> 1;                       // shift down to the next bit...
                CLR_CLK;
                }

        mask = 0x80;                                    // setup mask for MSB first
        for(i = 0; i < 8; i++)                  // send the 8 bits of data information
                {
                if((dac_data & mask) != 0)      // this bit is set???
                        {
                        SET_DATA;                               // set the data pin high
                        }
                else
                        {
                        CLR_DATA;                               // or set it low.
                        }
                SET_CLK;                                        // work the clock...
                mask = mask >> 1;                       // shift down to the next bit...
                CLR_CLK;
                }

        SET_CS;                                                 // latch in the data information
        }
```

Figure 12 – MAX512 Transfer Function

Test Results:
The logic pulse generator and synchronous serial transfer functionality of this project have been tested and verified for proper operation by the instructors of the intended course and graduate students within the department. The logic pulse generator is capable of providing the desired wave form from 10Hz through 100kHz. The duty cycle for the output wave form can be set from 0%, which is a continuous logic low signal, through 100%, which is a continuous logic high signal, in 1% increments.

The synchronous serial interface has been successfully utilized to communicate with a MAX512 DAC, a DS1267 digital potentiometer, a MAX1243 ADC, and a ML2036 DDS sine wave generator IC. The full project will be field tested by students enrolled in the course during the spring semester.

Future Work:

The next generation of the personal laboratory hardware will have a modified synchronous serial interface. The next version will require the user to effectively draw the desired timing diagram by selecting parameters to specify the type of chip select signal required along with the number of data bits, the order of the data bits (MSB or LSB first), and the clock polarity. This version of the interface will place more emphasis on the students understanding of the timing diagrams required to exchange data and will also provide more flexibility in the number and type of devices that can be utilized.

Additionally, the updates to the synchronous serial interface will replace the current ML2036 DDS sine wave generator IC with an updated version manufactured by Analog Devices.

Conclusions:

The overall project creates the hardware required to allow students enrolled in an advanced analog electronics course to perform laboratory experiments at their individual residence. This allows the students to be more prepared for their formal laboratory meetings and thus increase their productivity during laboratory meetings. The addition of the synchronous serial interface application allows the students to concentrate on the details of the devices being studied and not be limited by a lack of microcontroller or microcomputer understanding. This feature will allow the students to also improve their efficiency in the laboratory setting and increase the amount of material that can be covered since the students no longer need to generate software of their own to communicate with the synchronous devices.

References:

1 – Interview with J. Michael Jacobs, Purdue University, December 2009

2 – MAX512 Data Sheet, Retrieved from: www.maxim-ic.com

3 – DS1267 Data Sheet, Retrieved from: www.maxim-ic.com

4 – ML2036 Data Sheet, Retrieved from:
http://www2.tech.purdue.edu/ecet/courses/eet209/Lab_Boards/Lab_Boards.htm

5 – MAX1243 Data Sheet, Retrieved from: www.maxim-ic.com