HIT332: Embedded and Mobile Systems
Practical 4 Notes

Shane Reynolds

February 14, 2018

Contents

1 Introduction & Background

The intention behind this brief set of notes is to provide guidance on how well the practicals and projects for HIT332: Embedded and Mobile Systems achieve their intended outcomes. There are 5 practicals in total, and 3 projects. This set of notes will cover Practical 4. The practicals (and projects) make use of a development board created by Damien Hill and Ben Saunders of Charles Darwin University. The main component of the board is the Atmel ATmega1281 16au 16MHz, 8-bit microcontroller. The development board can be seen in Figure 1.

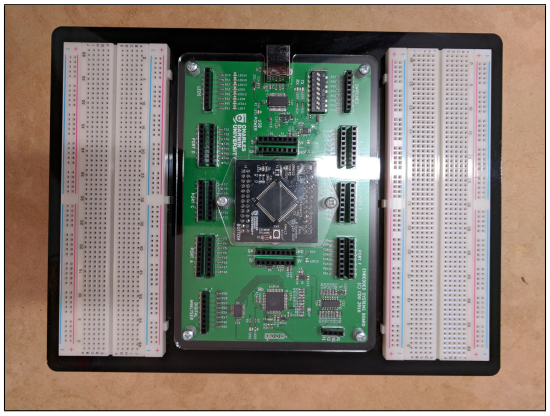

Figure 1: The development board which is used in practicals and projects for HIT332: Embedded and Mobile Systems

It must be highlighted that these notes have been developed using the board outside of its intended ecosystem. The board is made to be used on CDU's Casuarina Campus in one of the Engineering computer labs. These labs have the appropriate software installed in the correct file paths. These notes have been written using software installed on a personal machine which CDU does not control.

Furthermore, the components (other than the development board) used to complete the exercises were sourced independently from a local electronics supplier, independently of CDU. The notes will be highlighted where there has been a significant departure from the intended experience.

2 Schematic for the Simple Hardware Counter

This practical involves the creating a simple 4-bit binary counter, using both discrete hardware initially, and then implementing the counter using software. The 74LS193 is an IC which is the core of the 4-bit counter. It must be highlighted that the IC used for this practical was not supplied by CDU, and instead an identical device was acquired from a local electronics store. The IC is driven by a simple clock signal which is produced using software programmed into the ATmega1281 - essentially this will toggle a pin on and off. The schematics for the counter were created in KiCAD following the instructions in the Practical. The completed diagram can be seen in Figure 2.

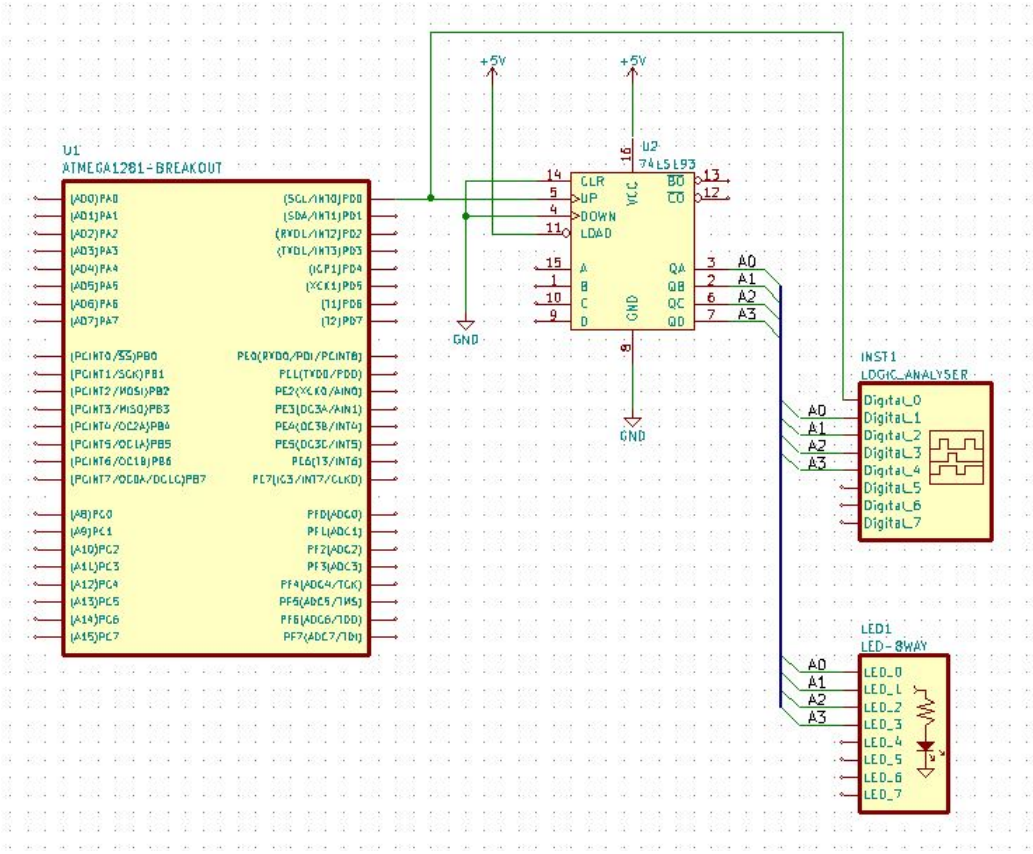

Figure 2: text

3 Clock Signal Code & Wiring the Development Board

A small piece of C code, `main.c`, was developed to generate the clock signal driving the 4-bit counter. This was compiled and loaded into the ATmega1281 with the same makefile process seen in Practical 3. Note that the modified make file, seen in Practical 3, was was copied into the Practical 4 directory. To compile and program the `avr_make all` and the `avr_program PORT=COM3` commands were run from the terminal, respectively. The `main.c` function which provides the clock signal can be seen in Figure 3.

```c
#include <avr/io.h>
#include <util/delay.h>

#define CLOCK_DELAY_MS 16.67

int main (void){
  DDRD = 0x01;      // Set PD0 as output
   while(1){
     PORTD |= (1<<PD0);     // Set PD0 high
      _delay_ms(CLOCK_DELAY_MS);
     PORTD &= ~(1<<PD0);    // Set PD0 low
      _delay_ms(CLOCK_DELAY_MS);
   }
}

```

Figure 3: text

Obviously, prior to programming the microcontroller, the development board was wired up according to the schematic shown in Figure 2. The actual physical implementation can be seen in Figure 4.

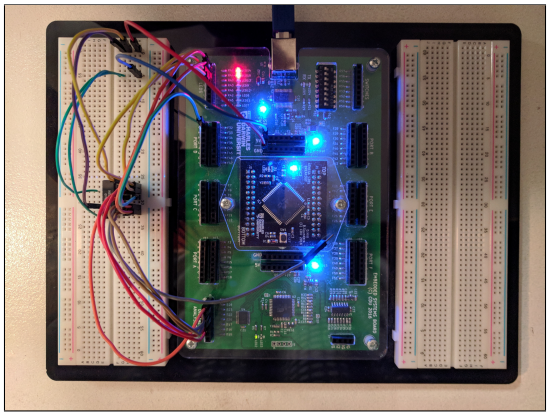

Figure 4: text

ATTENTION!!

The device did **not perform as expected**. Initially there was no response from the LED. To troubleshoot this issue, PD0 was connected to a single LED to test that it was toggling correctly - it was. Further testing involved checking the IC was grounded as required. Trial and error, while investigating the correct ground wiring, uncovered correct counter performance. To achieve this pin 4 on the IC needed to be disconnected from the ground. This may be caused by one of 2 possible factors:

1. The IC counter has been incorrectly wired for this practical;
2. The procured 74LS193 IC differs in some way to the IC which would typically be provided by CDU.

4 Using the Logic Counter

The scope was launched from the CDU Embedded Toolbox. The Digital Analyser was used to visualise the output from the 4-bit counter as per the instructions in the practical. This worked as expected, as shown in Figure 5. Note that the delay in the `main.c` implementation was set to 16.67m sec which is why the clock signal on channel 0 seems quick.

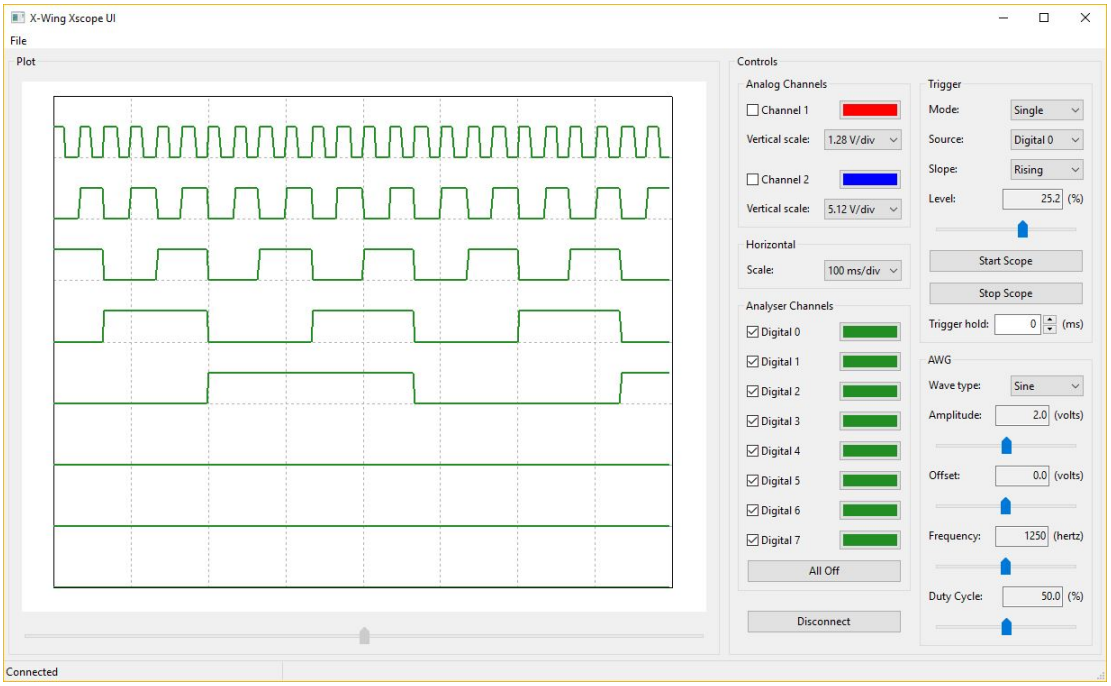

Figure 5: text

5 Creating a Software Counter

The final part of this practical sees the recreation of the 4-bit counter using only software. No KiCAD schematics were created for this implementation, however, the physical set up can be seen in Figure 6.

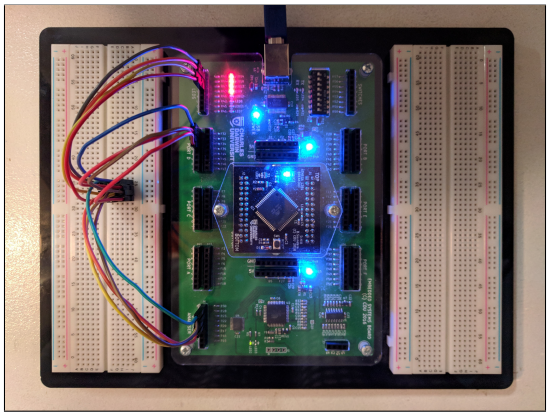

Figure 6: text

The hardware configuration for this implementation is a simple connection of PD0 to PD3 to LEDs, and the Digital Analyser channels. The C code implementation can be seen in Figure 7. Finally, the Digital Analyser output can be seen in Figure 8.

```c
#include <avr/io.h>
#include <util/delay.h>

#define CLOCK_DELAY_MS 50

int main (void){
unsigned char counter = 0;
DDRD = 0x0F;      // Set PD0 to PD3 (inclusive) as output
  while(1){
    counter++;
    if (counter > 15){
      counter = 0;
    }
    PORTD = counter;
    _delay_ms(2*CLOCK_DELAY_MS);
  }
}

```

Figure 7: text

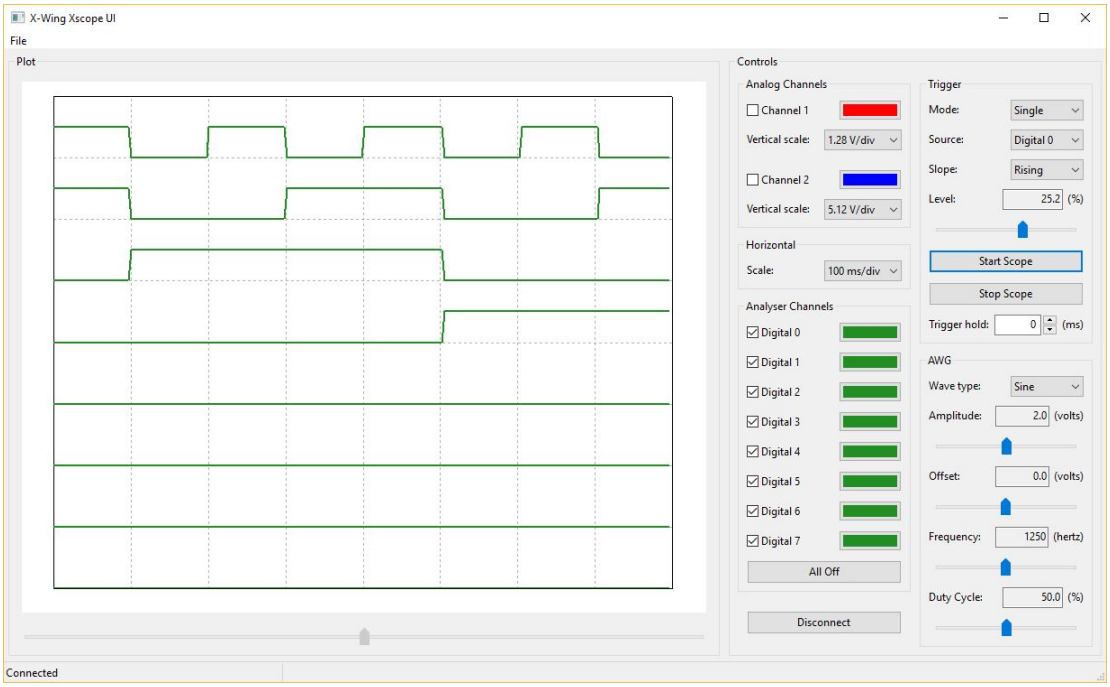

Figure 8: text