Demo 0: GPIO

ECE3056 Fall 2016

October 14, 2016

1 Overview

This program flashes both LEDs of the STM32 Value Line discovery board in a regular pattern, using simple delay loops to control timing.

2 Operation

When it is first reset, the STM32F100 microcontroller reads its initial program counter and stack pointer out of the interrupt vector table. [1] This vector table contains the addresses of code that is run in response to various *interrupt* events; external (or internal) triggers that will cause the processor to execute a special subroutine called a *handler*. In fact, the initial entry point on a new reset can be thought of as an interrupt handler for the most basic event of all: resetting the processor. Future demos will explore interrupts in further detail, but for now we simply use the interrupt vector table, stored at the very beginning of our program flash memory (address `0x08000000`) to provide the initial program counter and set up the stack pointer.

Our initial stack pointer is set to `0x20000400`, 1 kilobyte into the SRAM area, which starts at `0x20000000`. [2]

2.1 Memory-Mapped I/O

Some processor architectures, like the Intel x86 and Zilog Z80 lines, provide separate instructions for accessing memory and I/O devices. An instruction called `IN` or `OUT` would be used to read or write registers in I/O devices. Our ARM core, however, uses the same address space for memory and peripherals, so `LDR` and `STR` type instructions are used for accessing on-chip flash, SRAM, and peripherals. A complete map of its address space can be found on page 30 of the datasheet.

[1] PM0056 p. 36

[2] Datasheet p. 11

2.2 Using the GPIO Pins as Outputs

GPIO (general-purpose input and output) pins are perhaps the defining characteristic that distinguish microcontrollers from other systems-on-chip. The STM32F100 provides a large number of GPIO pins that can be configured as inputs, or outputs, with optional internal pull-ups or pull-downs, making it possible to directly connect them to switch contacts with no external resistors. Additionally, many of these pins have multiple functions, being configurable as inputs and outputs for ADCs, DACs, and PWM generators.

Each 16-wire I/O port is referred to by letter. The two LEDs on are board are connected to Port C, pins 8 and 9.[3] To configure these as outputs, we must first enable the clock signal using the RCC (reset and clock control) register space,[4] then set pins 8 and 9 as outputs using the GPIO port C register space. [5] Once these configuration steps have completed, bits 8 and 9 of any data written to the ODR (output data register) in the GPIO port C register space will appear on the LEDs.

3 Further Reading

The primary sources for all of the information used in this document are:

1. ST Microelectronics User Manual UM0919
   - The user manual for the STM32 Value Line Discovery board.
2. ST Microelectronics Reference Manual RM0041
   - The reference manual for the STM32f100RB microcontroller.
3. ST Microelectronics Programming Manual PM0056
   - Programming manual for ST Micro's Cortex M3-based microcontrollers.
4. ST Microelectronics STM32F100RB datasheet
   - Datasheet for the part; contains pinouts and memory map.

These are readily available in PDF format online.

4 Exercises

The following exercises can be completed by modifying the demo code.

1. Make the LEDs flash once, then twice, then three times, etc.

[3]UM0919 p. 11
[4]RM0041 p. 90
[5]RM0041 p. 111

2. Instead of flashing both simulataneously, modify the code to alternate between the two LEDs.
3. Modify the flashing of the LED to make it faster than the human eye can perceive. Look at the signal appearing on the `PC9` pin header on an oscilloscope.