CSE 320 **Fall 2016**

Computer Project #11

Assignment Overview

This assignment develops familiarity with the instruction set architecture of the ARM microprocessor.

It is worth 30 points (3% of course grade) and must be completed no later than 11:59 PM on Thursday, 12/8.

Assignment Deliverables

The deliverables for this assignment are the following files:

`proj11.makefile` – the makefile which produces "proj11"
`proj11.support.c` – the source code for your support module
`proj11.driver.c` – the source code for your driver module

Be sure to use the specified file names and to submit them for grading via the CSE handin system.

Assignment Specifications

The program will simulate some of the actions taken during the fetch-execute cycle for the ARM microprocessor.

1. You will develop a support module which calculates the value which should be used to update the PC register at the end of the execute stage in the fetch-execute cycle. The interface to the support module is through function "update", which is declared as follows:

```
unsigned update( unsigned PC, unsigned IR, unsigned CPSR );
```

The first argument is the current contents of the PC register, the second argument is the current contents of the IR register, and the third argument is the current contents of the CPSR register.

Based on that machine language instruction in the IR register, function "update" will compute and return the next value of the PC. The value zero will be returned for illegal machine language instructions.

The support module will consist of function "update" and any additional helper functions which you choose to implement. The support module will not perform any input or output operations. For example, the functions in the support module will not call function "scanf" or function "printf".

2. You will develop a driver module to test your implementation of the support module. The driver module will consist of function "main" and any additional helper functions which you choose to implement. All output will be appropriately labeled.

Your driver module may not be written as an interactive program, where the user supplies input in response to prompts. Instead, your test cases will be included in the source code as literal constants.

Assignment Notes

1. Your driver module and your support module must be in separate source code files.

2. Your source code must be translated by "gcc", which is a C compiler and accepts C source statements.

3. You must supply a "makefile" (named "proj11.makefile"), and that makefile must produce an executable program named "proj11".

4. Note that the functions in your support module cannot perform any input or output operations. All communication between the driver module and the support module will be done via the three arguments to function "update".

5. This project will focus on a subset of the ARM machine language instructions. Bits 27:26 of the IR identify the category to which a machine language instruction belongs:

```
00  Data Processing
01  Data Movement
10  Branch
```

6. The following describes the format and actions performed by Branch instructions.

```
Bits 31:28     cond (condition field)
Bits 27:26     10
Bits 25:24     1L (L is 0 for B, L is 1 for BL)
Bits 23:0      simm24 (signed immediate 24-bit value)
```

When the condition field indicates that the branch should be taken, the following computation is used:

```
PC + sign_extend( simm24 << 2 ) → PC
```

Otherwise, the default computation for all instructions is used:

```
PC + 4 → PC
```

7. The following table lists the condition field entries.

cond	meaning	NZCV state
0000	equal	Z set
0001	not equal	Z clear
0010	carry	C set
0011	not carry	C clear
0100	negative	N set
0101	not negative	N clear
0110	overflow	V set
0111	not overflow	V clear
1000	unsigned greater than	C set and Z clear
1001	unsigned less than or equal to	C clear or Z set
1010	signed greater than or equal to	N == V
1011	signed less than	N != V
1100	signed greater than	Z clear and N == V
1101	signed less than or equal to	Z set or N != V
1110	always	irrelevant

8. Machine language instructions which do not meet the criteria given above are to be processed as illegal instructions.