Lab 06

R. Ferrero, A. C. Marceddu

Politecnico di Torino

Dipartimento di Automatica e Informatica (DAUIN)

Torino - Italy

Exercise 1

- Rename register `r1` to `single_value`, `r2` to `double_value`, `r3` to `triple_value`, `r4` to `quadruple_value` e `r5` to `quintuple_value`
- Assign some value to `single_value`

Exercise 1 (cont.)

- By only using `MOV` and `ADD`, assign these values to the registers:
  - `double_value` = `single_value` *2
  - `triple_value` = `single_value` *3
  - `quadruple_value` = `single_value` * 4
  - `quintuple_value` = `single_value` * 5
- Suggestion: exploit Inline Barrel Shifter with `MOV`

Exercise 2

- Allocate 26 byte into a memory area DATA READWRITE, without initializing them
- Initialize `r0` and `r1` to 1
- Assign to registers `r2-r12` the elements of Fibonacci sequence. For example:
  - `r2 = r1 + r0`
  - `r3 = r2 + r1`
- Assign to `r14` the address of the first byte of memory area allocated before

Exercise 2 (cont.)

- Using pre-indexed addressing, save the least significant byte of registers `r0-r12`, incrementing `r14` at each assingment
- Using post-indexed addressing mode, save the least significant byte of registers `r12-r0` (reverse order), incrementing `r14` at each assignment.
- At the end, check that the content of the memory is the following:

01 01 02 03 05 08 0D 15 22 37 59 90 E9 E9 90 59 37 22 15 0D 08 05 03 02 01 01

Exercise 3

Write a program in ARM assembly language executing this tasks:

- Define the following constants in a code area

```
myConstants DCW 57721,56649, 15328,
 60606, 51209, 8240, 24310, 42159
```

- Allocate 16 byte (4 word) in a data area
- Considering the constants into couples, write in the 4 word the sum of the 4 couples of constants.

Exercise 4

- Cortex-M4 instruction set contains the following instruction:

```
UADD8 <Rd>, <Rn>, <Rm>
```

- `UADD8` sums corresponding bytes of `Rn` and `Rm`, storing the result in `Rd`.
- Example: `Rn` = 0x 7A 30 45 8D

  `Rm` = 0x C3 15 9E AA

  `Rd` = 0x 3D 45 E3 37
- Please note the absence of carry between bytes in `Rd`.

Exercise 4 (cont.)

- `UADD8` is not present in Cortex-M3 instruction set.
- Write instructions for Cortex-M3 equivalent to `UADD8 r4, r0, r1`.

Exercise 5

- Cortex-M4 instruction set contains the following instruction:

  ```
  USAD8 <Rd>, <Rn>, <Rm>
  ```

- Each byte of `Rn` and `Rm` is a pure binary number.
- `USAD8` calculates the absolute value of the difference between each byte in `Rn` and `Rm`.
- After that, `USAD8` sums the four absolute values, storing the result in in `Rd`.

Exercise 5 (cont.)

- Example: `Rn` = 0x 7A 30 45 8D

  `Rm` = 0x C3 15 9E AA

1. | 0x8D - 0xAA| = 0x1D
2. | 0x45 - 0x9E | = 0x59
3. | 0x30 - 0x15 | = 0x1B
4. | 0x7A – 0xC3 | = 0x49

`Rd` = 0x1D + 0x59 + 0x1B + 0x49 = 0xDA

- Note: the value in `Rd` can be on more than 8 bit

Exercise 5 (cont.)

- `USAD8` is not present in Cortex-M3 instruction set.
- Write instructions for Cortex-M3 equivalent to `USAD8 r5, r0, r1`.

Exercise 6

- Cortex-M4 instruction set contains the following instructions:

```
SMUAD <Rd>, <Rn>, <Rm>
SMUSD <Rd>, <Rn>, <Rm>
```

- Both instructions multiply the lower halfword of `Rn` times the lower halfword of `Rm`, and the higher halfword of `Rn` times the higher halfword of `Rm`.

Exercise 6 (cont.)

- Halfwords are considered in two's complement.
- `SMUAD` sums the two products and saves the result in `Rd`.
- `SMUSD` subtracts the product of the higher halfwords from the product of the lower halfwords, storing the result in `Rd`.

Exercise 6 (cont.)

- Example: `Rn` = 0x7A30 458D
  `Rm` = 0xC315 9EAA
- 0x458D * 0x9EAA = 0xE58E35A2
- 0x7A30 * 0xC315 = 0xE2EC95F0
- With `SMUAD`, `Rd` = 0xC87ACB92
- With `SMUSD`, `Rd` = 0x02A19FB2

Exercise 6 (cont.)

- `SMUAD` e `SMUSD` are not present in the Cortex-M3 instruction set.
- Write instructions for Cortex-M3 equivalent to

```
SMUAD r6, r0, r1
SMUSD r7, r0, r1
```

Exercise 6 (cont.)

- The sign of halfwords has to be extended before multiplication.
- Example in pure binary:
  - 0x458D = 17805
  - 0x9EAA = 40618
  - 0x458D * 0x9EAA = 0x2B1B35A2 = 723.203.490
- In two's complement:
  - 0x9EAA = -24918
  - 0x458D * 0x9EAA = 0xE58235A2 = -443.664.990

Exercise 7 (optional)

- Create a new project by selecting a card with the Cortex-M4 core, for example NXP LPC4072. Write the instructions:

```
UADD8 r4, r0, r1
USAD8 r5, r0, r1
SMUAD r6, r0, r1
SMUSD r7, r0, r1
```

- Verify that results are coherent with the ones obtained in the previous exercises.