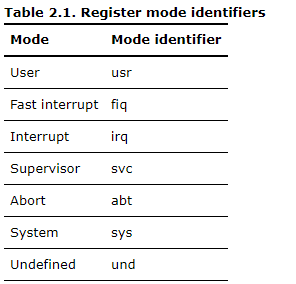
HW#2

1. The ARM7TDMI has 7 different modes listed in in the following table found on <http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0210c/Cihhcjia.html>



It has two States, ARM vs Thumb arm being a 32-bit state while Thumb being a 16-bit state

1. Generally, r13 is the stack register which points to a stack, an area of allocated memory that can store data. For example in a program that uses multiple branches and for whatever the reason you wanted to save the data in each current register you can push the data of each register onto the stack in a specific order (ascending, descending). And retrieve that data later on. R13 also referred to as SP contains a pointer to the last entry of the stack so that the data storage can be stored to that location or retrieved from that location.

R14, is the link register also called LR stores the return address, after a branch link instruction the address of where the program branched is stored in the link register, it is updated after every branch if multiple branches are used the link register must be stored in a stack to preserve its data.

R15, is the program counter also called PC, after each instruction is fetched decoded and executed the program counter is advanced, it keeps track of how long the program has been running

1. The first four bits of the status registers sets the flags ZCNV
2. When a program encounters undefined mode it will jump to 0x00000004 of an external memory according to p64 of the book
3. You bit clear the SPSR which contains the current value of CPSR with 0x80 which will set this value the 8th bit from the right to 0
4. There are three stages fetch decode and execute.
5. CPSR = 0xF000 00D3 = 1111(F) …. 1101(D) 0011(3)

From this all the flags NZVC are set to 1

110 – bits [7:5] represent I F and T since I and F are both set to 1 then the IRQ and FIQ are disabled and since T is 0 the processor is in the ARM state

The rest of the bits [4:0] are used to determine the mode 10011 correponds with supervisor mode.

AREA Prog1, CODE, READONLY

ENTRY

mov r1, x ;not a actual instruction just assuming x is on r1

3X\_squared

mul r2, r1, r1 ;stored x squared into r2 while perserving values in r1

mul r2, #3 ;this could be used to find 3x^2

negX\_2

mul r1, r1, #2 ;multiplies r1(x) with 2 and stores it back to r1

plus\_6

mov r3, #6 ;I am now going to sum up all the values into r3 with an initial

;condition of 6 this will represent +6

subs r2, r2, r1 ;if 3x^2 is less then 2x then the N flag is set

add r3, r3, r3 ;either way it still does not change the answer in r3

stop B stop ; stop program

END

