CS1021 Tutorial 2

Q1 The NXP LPC2468 has 512KiB of flash memory starting at address 0x00000000. What is the last address of the memory area (in hexadecimal)?

512KiB = 512 x 1024 = 524,288 bytes

Convert 524,288 to hexadecimal

\[
\begin{array}{c|c}
  16 & 524,288 \\
  16 & 32,768 \quad r \ 0 \\
  16 & 2,048 \quad r \ 0 \\
  16 & 128 \quad r \ 0 \\
  16 & 8 \quad r \ 0 \\
  16 & 0 \quad r \ 8 \\
\end{array}
\]

524,288 = 0x00080000

Last address of memory area is 0x00000000 + 0x00080000 – 1 = 0x0007FFFF

Need to subtract 1 because first byte at offset 0 (not 1).

Q2 The NXP LPC2468 has 64KiB of read write memory starting at address 0x40000000. What is the last address of the memory area (in hexadecimal)?

Q3 One hundred 8 bit unsigned integers are stored consecutively in memory starting at address 0x00002000. What is the address of the byte containing (i) the first integer (ii) the 22\textsuperscript{nd} integer (iii) the 75\textsuperscript{th} integer and (iv) the last integer?

Q4 One hundred 32 bit signed integers are stored consecutively in memory starting at address 0x004420C0. What is the address of the word containing (i) the first integer (ii) the 22\textsuperscript{nd} integer (iii) the 75\textsuperscript{th} integer and (iv) the last integer?

The first integer is stored at 0x00442240. As each integer is 32 bits or 4 bytes, the address of the 2\textsuperscript{nd} integer is 0x004420C0 + 4 = 0x004420C4, and so on. The n\textsuperscript{th} integer is at address 0x004420C0 + 4(n-1). If n = 100, the 100\textsuperscript{th} integer is at address 0x004420C0 +
4 * 99 = 0x004420c0 + 0x018c = 0x00442224C. The addresses of the 22\textsuperscript{nd} (0x00442114) and 75\textsuperscript{th} (0x004421E8) integers can be computed in a similar way.

Q5 Assuming that x is stored in R1, y in R2, z in R3 and the result in R0:

(i) Write ARM assembly language instructions to compute x + y + z.

\[
\begin{align*}
\text{ADD} & \quad \text{R0, R1, R2} \quad ; \ R0 = x + y \\
\text{ADD} & \quad \text{R0, R0, R3} \quad ; \ R0 = x + y + z
\end{align*}
\]

(ii) Write ARM assembly language instructions to compute y - x - z.

(iii) Write ARM assembly language instructions to compute x\(^2\) + y\(^2\) + z\(^2\).

(iv) Write ARM assembly language instructions to compute 5(x + y).

\[
\begin{align*}
\text{MOV} & \quad \text{R0, #5} \quad ; \ R0 = 5 \\
\text{ADD} & \quad \text{R4, R1, R2} \quad ; \ R4 = x + y \\
\text{MUL} & \quad \text{R0, R4, R0} \quad ; \ R0 = 5^\ast(x + y)
\end{align*}
\]

Need to work around the limitations of the MUL instruction - dst and src1 registers must not be the same and src2 cannot be an immediate value.

(v) Write ARM assembly language instructions to compute (x + y)(y - z).

(vi) Write ARM assembly language instructions to compute 3x\(^4\) - 5x - 16y\(^4\)z\(^4\).

\[
\begin{align*}
\text{MOV} & \quad \text{R0, #3} \quad ; \ R0 = 3 \\
\text{MUL} & \quad \text{R4, R1, R1} \quad ; \ R1 = x^2 \\
\text{MUL} & \quad \text{R0, R4, R0} \quad ; \ R4 = 3x^2 \\
\text{MUL} & \quad \text{R0, R4, R0} \quad ; \ R0 = 3x^4 \\
\text{MOV} & \quad \text{R4, #5} \quad ; \ R4 = 5 \\
\text{MUL} & \quad \text{R4, R1, R4} \quad ; \ R4 = 5x \\
\text{SUB} & \quad \text{R0, R0, R4} \quad ; \ R0 = 3x^4 - 5x \\
\text{ADD} & \quad \text{R4, R2, R2} \quad ; \ R4 = 2y \\
\text{MUL} & \quad \text{R4, R3, R4} \quad ; \ R4 = 2yz \\
\text{MUL} & \quad \text{R5, R4, R4} \quad ; \ R5 = 4y^2z^2 \\
\text{MUL} & \quad \text{R4, R5, R5} \quad ; \ R5 = 16y^4z^4 \\
\text{SUB} & \quad \text{R0, R0, R4} \quad ; \ R0 = 3x^4 - 5x - 16y^4z^4
\end{align*}
\]

Need to work around the limitations of the MUL instruction - dst and src1 registers must not be the same and src2 cannot be an immediate value.