# Computer Architecture Assignment #1

Your Student Number: 2021320050

Name: 장오선/Zhang Aoxuan

<<Notice>>

- Please edit the title of this document correctly.
    - (ex. CA1_2022012345_Tom-Cruise OR CA1_2022012345_홍길동)
- Please write your information(Student number and name) correctly.
- You can write your answers in English or Korean.
- Don't change the format(colored in white) of this document. (Please edit just the blue-colored part.)

**Address 000 | Instruction EA000006 (Example)**

a) Change to binary format: 1110 1010 0000 0000 0000 0000 0000 0110
b) Write assembly code: B #8;
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Branch and branch with the link' is only one instruction set encoding whose value at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
    b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn't need to store any return address because the L bit is 0.
    d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
        i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 00 0000 0000 0000 0000 0000 0000 0110. (Because the signed immediate is 0000 0000 0000 0000 0000 0110 here.)
        ii. Then, get 0000 0000 0000 0000 0000 0000 0001 1000 by shifting the result left two bits.
        iii. Because the address of this instruction is 0, the content of the PC will be 0 + 8 bytes. So, the target address will be (0+8) + 24 = 32(bytes). It means after the operation of this instruction, the PC will be moved to 32/4 = 8.
        iv. Therefore, I can write the assembly code of this instruction like 'B #8;' because the syntax of the branch instruction is 'B{L}{cond} <target_address>'.
d) What is the meaning of the instruction? : The instruction means 'branch to address 8'.

Address 001 | Instruction EAFFFFFE

a) Change to binary format: 1110 1010 1111 1111 1111 1111 1111 1110
b) Write assembly code: B #1;
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Branch and branch with the link' is only one instruction set encoding whose value at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
    b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn't need to store any return address because the L bit is 0.
    d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
        i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 11 1111 1111 1111 1111 1111 1111 1110. (Because the signed immediate is 1111 1111 1111 1111 1111 1111 1110 here.)
        ii. Then, get 1111 1111 1111 1111 1111 1111 1111 1000 by shifting the result left two bits.
        iii. Because the address of this instruction is 1, the content of the PC will be 1*4 + 8 bytes. So, the target address will be (1*4+8) - 8 = 4(bytes). It means after the operation of this instruction, the PC will be moved to 4/4 = 1.
        iv. Therefore, I can write the assembly code of this instruction like 'B #1;' because the syntax of the branch instruction is 'B{L}{cond} <target_address>'.
d) What is the meaning of the instruction? : The instruction means 'branch to address 1'.

Address 002 | Instruction EA0000A7

a) Change to binary format: 1110 1010 0000 0000 0000 0000 1010 0111
b) Write assembly code: B #171;
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Branch and branch with the link' is only one instruction set encoding whose value at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
    b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn't need to store any return address because the L bit is 0.

d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
    i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 00 0000 0000 0000 0000 0000 1010 0111. (Because the signed immediate is 0000 0000 0000 0000 1010 0111 here.)
    ii. Then, get 0000 0000 0000 0000 0000 0010 1001 1100 by shifting the result left two bits.
    iii. Because the address of this instruction is 2, the content of the PC will be 2*4 + 8 bytes. So, the target address will be (2*4+8) + 668 = 684(bytes). It means after the operation of this instruction, the PC will be moved to 684/4 = 171.
    iv. Therefore, I can write the assembly code of this instruction like 'B #171;' because the syntax of the branch instruction is 'B{L}{cond} <target_address>'.

d) What is the meaning of the instruction? : The instruction means 'branch to address 171'.

Address 003~005 | Instruction EAFFFFFE

a) Change to binary format: 1110 1010 1111 1111 1111 1111 1111 1110
b) Write assembly code: 003: B #3;  004: B #4;  005: 'B #5;
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Branch and branch with link' is only one instruction set encoding whose value at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
    b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn't need to store any return address because the L bit is 0.
    d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
        i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 11 1111 1111 1111 1111 1111 1111 1110. (Because the signed immediate is 1111 1111 1111 1111 1111 1111 1110 here.)
        ii. Then, get 1111 1111 1111 1111 1111 1111 1111 1000 by shifting the result left two bits.
        iii. Because the address of this instruction is from 3 to 5, the content of the PC will be 3*4 + 8, 4*4 + 8, and 5*4+ 8 bytes, respectively. So, for the address 3, the target address will be (3*4 + 8) - 8 = 12(bytes). It means after the operation of this instruction, the PC will be moved to 12/4 = 3. For the address 4, the target address will be (4*4 + 8) - 8 = 16(bytes). It means after the operation of this instruction, the PC will be moved to 16/4 = 4, and for the address 5, the target address will be (5*4 + 8) - 8 = 20(bytes). It means after the operation of this instruction, the PC will be moved to 20/4 = 5.

iv. Therefore, I can write the assembly code of this instruction like ‘B #3;’, ‘B #4;’, and ‘B #5;’ respectively, because the syntax of the branch instruction is ‘B{L}{cond} <target_address>’.

d) What is the meaning of the instruction? : The instruction in address 3 means ‘branch to address 3’, the instruction in address 4 means ‘branch to address 4’, and the instruction in address 5 means‘ branch to address 5’.

Address 006 | Instruction EA0000A4

a) Change to binary format: 1110 1010 0000 0000 0000 0000 1010 0100
b) Write assembly code: B #172;
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, ‘Branch and branch with link’ is only one instruction set encoding whose value at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
    b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The ‘Operation’ part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn’t need to store any return address because the L bit is 0.
    d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
        i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 00 0000 0000 0000 0000 0000 1010 0100. (Because the signed immediate is 0000 0000 0000 0000 1010 0100 here.)
        ii. Then, get 0000 0000 0000 0000 0000 0010 1001 0000 by shifting the result left two bits.
        iii. Because the address of this instruction is 6, the content of the PC will be 6*4 + 8 bytes. So, the target address will be (6*4+8) + 656 = 688(bytes). It means after the operation of this instruction, the PC will be moved to 688/4 = 172.
        iv. Therefore, I can write the assembly code of this instruction like ‘B #172;’ because the syntax of the branch instruction is ‘B{L}{cond} <target_address>’.
d) What is the meaning of the instruction? : The instruction means ‘branch to address 172’.

Address 007 | Instruction EAFFFFFE

a) Change to binary format: 1110 1010 1111 1111 1111 1111 1111 1110
b) Write assembly code: B #7;
c) Describe why you wrote the assembly code like the above:

a. Type of instruction: According to figure A3-1 in the ARM manual, 'Branch and branch with link' is only one instruction set encoding whose value at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn't need to store any return address because the L bit is 0.
d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
    i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 11 1111 1111 1111 1111 1111 1111 1110. (Because the signed immediate is 1111 1111 1111 1111 1111 1111 1110 here.)
    ii. Then, get 1111 1111 1111 1111 1111 1111 1111 1000 by shifting the result left two bits.
    iii. Because the address of this instruction is 7, the content of the PC will be 7*4 + 8 bytes. So, the target address will be (7*4+8) - 8 = 28(bytes). It means after the operation of this instruction, the PC will be moved to 28/4 = 7.
    iv. Therefore, I can write the assembly code of this instruction like 'B #7;' because the syntax of the branch instruction is 'B{L}{cond} <target_address>'.

d) What is the meaning of the instruction? : The instruction means 'branch to address 7'.

Address 008 | Instruction E59F2EC8

a) Change to binary format: 1110 0101 1001 1111 0010 1110 1100 1000
b) Write assembly code: LDR $2, [$15, #0xEC8];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0010 in binary and means Rd($2) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1111($15), and addr_mode is 1110 1100 1000.

ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
We have P == 1 and W == 0, the base register is not updated. (offset addressing)
The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 1, meaning that this operation will be a Load. Hence the operation has to be Load Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($15) containing the base address and the immediate offset(#0xEC8) used with the value of Rn to form the address, that is [$15, #+EC8]

iii. Therefore, I can write the assembly code of this instruction like 'LDR $2, [$15, #0xEC8];' because the syntax of load instruction is 'LDR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 15th register with #0xEC8. Read the value saved in the address [$15 + #0xEC8] of memory and load it to the second register $2.

Address 009 | Instruction E3A00040

a) Change to binary format: 1110 0011 1010 0000 0000 0000 0100 0000
b) Write assembly code: MOV $0 #0x40
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual and the bit[26:27], it should be a 'Data Processing/PSR Transfer' operation, and continuously looking up the code at [21:24], that is 1101, which is represented as opcode, in accordance with the A3-7 in ARM manual, so I can figure out is it a Move instruction(A4-68).
    b. Operation – Condition Field: According to A3.4(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – I: Distinguishes between the immediate and register forms of <shifter_operand>. Here I == 1, which shows that operand 2 is an immediate value.
    d. Operation – opcode: 1101 = MOV - Rd:= Op2
    e. Operation – S: 0 indicates it does not alter condition codes
    f. Operation – Rn: 0000, that is the first operand register
    g. Operation – Rd: 0000, that is the destination register
    h. Operation – shifter_operand:
        i. [11:8]: 0000, shift applied to Imm
        ii. [7:0]: 0100 0000, unsigned 8-bit immediate value, 40 in hexadecimal
d) What is the meaning of the instruction? : Copies the value of 0x40 into the zeroth register.

Address 00A | Instruction E5820010

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0001 0000

b) Write assembly code: STR $0, [$2, #0x010];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Load Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0001 0000.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing)
        The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x010) used with the value of Rn to form the address, that is [$2, #+010]
        iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x010];' because the syntax of the store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.
d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x010. Read the value saved in $0 and store it in the address [$2 + #0x010] of memory.

Address 00B | Instruction E5820014

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0001 0100
b) Write assembly code: STR $0, [$2, #0x014];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check

the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)

c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
    i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0001 0100.
    ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
    We have P == 1 and W == 0, the base register is not updated. (offset addressing)
    The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] which represents the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x014) used with the value of Rn to form the address, that is [$2, #+014]
    iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x014];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x014. Read the value saved in $0 and store it in the address [$2 + #0x014] of memory.

Address 00C | Instruction E5820018

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0001 1000
b) Write assembly code: STR $0, [$2, #0x018];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.

i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0001 1000.
ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
We have P == 1 and W == 0, the base register is not updated. (offset addressing)
The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x018) used with the value of Rn to form the address, that is [$2, #+018]
iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x018];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x018. Read the value saved in $0 and store it in the address [$2 + #0x018] of memory.

Address 00D | Instruction E582001C

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0001 1100
b) Write assembly code: STR $0, [$2, #0x01C];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0001 1100.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing)
        The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation

will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x01C) used with the value of Rn to form the address, that is [$2, #+01C]

iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x01C];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x01C. Read the value saved in $0 and store it in the address [$2 + #0x01C] of memory.

Address 00E | Instruction E5820020

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0010 0000
b) Write assembly code: STR $0, [$2, #0x020];
c) Describe why you wrote the assembly code like the above:
   a. Type of instruction: According to figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
   b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
   c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
   d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
      i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0010 0000.
      ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
      We have P == 1 and W == 0, the base register is not updated. (offset addressing)
      The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x020) used with the value of Rn to form the address, that is [$2, #+020]
      iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x020];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x020. Read the value saved in $0 and store it in the address [$2 + #0x020] of memory.

Address 00F | Instruction E5820024

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0010 0100
b) Write assembly code: STR $0, [$2, #0x024];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0010 0100.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing) The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x024) used with the value of Rn to form the address, that is [$2, #+024]
        iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x024];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.
d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x024. Read the value saved in $0 and store it in the address [$2 + #0x024] of memory.

**Address 010 | Instruction E3A0003F**

a) Change to binary format: 1110 0011 1010 0000 0000 0000 0011 1111

b) Write assembly code: MOV $0 #0x3F
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual and the bit[26:27], it should be a 'Data Processing/PSR Transfer' operation, and continuously looking up the code at [21:24], that is 1101, which is represented as opcode, in accordance with the A3-7 in ARM manual, so I can figure out is it a Move instruction(A4-68).
    b. Operation – Condition Field: According to A3.4(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – I: Distinguishes between the immediate and register forms of <shifter_operand>. Here I == 1, which shows that operand 2 is an immediate value.
    d. Operation – opcode: 1101 = MOV - Rd:= Op2
    e. Operation – S: 0 indicates it does not alter condition code
    f. Operation – Rn: 0000, that is the first operand register
    g. Operation – Rd: 0000, that is the destination register
    h. Operation – shifter_operand:
        i. [11:8]: 0000, shift applied to Imm
        ii. [7:0]: 0011 1111, unsigned 8-bit immediate value, 3F in hexadecimal
d) What is the meaning of the instruction? : Copies the value of 0x3F into the zeroth register.

Address 011 | Instruction E5820028

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0010 1000
b) Write assembly code: STR $0, [$2, #0x028];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0010 1000.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing)

The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x028) used with the value of Rn to form the address, that is [$2, #+028]

iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x028];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x028. Read the value saved in $0 and store it in the address [$2 + #0x028] of memory.

Address 012 | Instruction E3A00008

a) Change to binary format: 1110 0011 1010 0000 0000 0000 0000 1000
b) Write assembly code: MOV $0 #0x08
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual and the bit[26:27], it should be a 'Data Processing/PSR Transfer' operation, and continuously looking up the code at [21:24], that is 1101, which is represented as opcode, by the A3-7 in ARM manual, so I can figure out is it a Move instruction(A4-68).
    b. Operation – Condition Field: According to A3.4(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – I: Distinguishes between the immediate and register forms of <shifter_operand>. Here I == 1, which shows that operand 2 is an immediate value.
    d. Operation – opcode: 1101 = MOV - Rd:= Op2
    e. Operation – S: 0 indicates it does not alter condition code
    f. Operation – Rn: 0000, that is the first operand register
    g. Operation – Rd: 0000, that is the destination register
    h. Operation – shifter_operand:
        i. [11:8]: 0000, shift applied to Imm
        ii. [7:0]: 0000 1000, unsigned 8-bit immediate value, 8 in hexadecimal
d) What is the meaning of the instruction? : Copies the value of 0x08 into the zeroth register.

Address 013 | Instruction E582002C

a) Change to binary format: 1110 0101 1000 0010 0000 0000 0010 1100
b) Write assembly code: STR $0, [$2, #0x02C];
c) Describe why you wrote the assembly code like the above:

a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0000 in binary and means Rd($0) for the loaded value.
d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
    i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0010($2), and addr_mode is 0000 0010 1100.
    ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
    We have P == 1 and W == 0, the base register is not updated. (offset addressing)
    The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($2) containing the base address and the immediate offset(#0x02C) used with the value of Rn to form the address, that is [$2, #+02C]
    iii. Therefore, I can write the assembly code of this instruction like 'STR $0, [$2, #0x02C];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 2nd register with #0x02C. Read the value saved in $0 and store it in the address [$2 + #0x02C] of memory.

Address 014 | Instruction E59F3E9C

a) Change to binary format: 1110 0101 1001 1111 0011 1110 1001 1100
b) Write assembly code: LDR $3, [$15, #0xE9C];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)

c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0011 in binary and means Rd($3) for the loaded value.
d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
    i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1111($15), and addr_mode is 1110 1001 1100.
    ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
    We have P == 1 and W == 0, the base register is not updated. (offset addressing)
    The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 1, meaning that this operation will be a Load. Hence the operation has to be Load Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($15) containing the base address and the immediate offset(#0xE9C) used with the value of Rn to form the address, that is [$15, #+E9C]
    iii. Therefore, I can write the assembly code of this instruction like 'LDR $3, [$15, #0xE9C];' because the syntax of load instruction is 'LDR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 15th register with #0xE9C. Read the value saved in the memory address [$15 + #0xE9C] and load it to the third register $3.

Address 015 | Instruction E59F1E9C

a) Change to binary format: 1110 0101 1001 1111 0001 1110 1001 1100
b) Write assembly code: LDR $1, [$15, #0xE9C];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0001 in binary and means Rd($1) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1111($15), and addr_mode is 1110 1001 1100.

ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
We have P == 1 and W == 0, the base register is not updated. (offset addressing)
The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 1, meaning that this operation will be a Load. Hence the operation has to be Load Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($15) containing the base address and the immediate offset(#0xE9C) used with the value of Rn to form the address, that is [$15, #+E9C]

iii. Therefore, I can write the assembly code of this instruction like 'LDR $1, [$15, #0xE9C];' because the syntax of load instruction is 'LDR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 15th register with #0xE9C. Read the value saved in the memory address [$15 + #0xE9C] and load it to the first register $1.

Address 016 | Instruction E5831000

a) Change to binary format: 1110 0101 1000 0011 0001 0000 0000 0000
b) Write assembly code: STR $1, [$3, #0x000];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0001 in binary and means Rd($1) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0011($3), and addr_mode is 0000 0000 0000.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing)
        The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($3) containing the

base address and the immediate offset(#0x000) used with the value of Rn to form the address, that is [$3, #+000]

iii. Therefore, I can write the assembly code of this instruction like 'STR $1, [$3, #0x000];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 3rd register with #0x000. Read the value saved in $1 and store it in the address [$3 + #0x000] of memory.

Address 017 | Instruction E59F9E98

a) Change to binary format: 1110 0101 1001 1111 1001 1110 1001 1000
b) Write assembly code: LDR $9, [$15, #0xE98];
c) Describe why you wrote the assembly code like the above: YOUR ANSWER
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 1001 in binary and means Rd($9) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1111($15), and addr_mode is 1110 1001 1000.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing) The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Load. Hence the operation has to be Load Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($15) containing the base address and the immediate offset(#0xE98) used with the value of Rn to form the address, that is [$15, #+E98]
        iii. Therefore, I can write the assembly code of this instruction like 'LDR $9, [$15, #0xE98];' because the syntax of load instruction is 'LDR{<cond>} <Rd>, <addr_mode>'.
d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 15th register with #0xE98. Read the value saved in the memory address [$15 + #0xE98] and load it to the ninth register $9.

Address 018 | Instruction E3A08000

a) Change to binary format: 1110 0011 1010 0000 1000 0000 0000 0000
b) Write assembly code: MOV $8 #0x00;
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual and the bit[26:27], it should be a 'Data Processing/PSR Transfer' operation, and continuously looking up the code at [21:24], that is 1101, which is represented as opcode, in accordance with the A3-7 in ARM manual, so I can figure out is it a Move instruction(A4-68).
    b. Operation – Condition Field: According to A3.4(Page A4-10), there is the detail of the Data Processing instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – I: Distinguishes between the immediate and register forms of <shifter_operand>. Here I == 1, which shows that operand 2 is an immediate value.
    d. Operation – opcode: 1101 = MOV - Rd:= Op2
    e. Operation – S: 0 indicates it does not alter condition code
    f. Operation – Rn: 0000, that is first operand register $0
    g. Operation – Rd: 1000, that is destination register $8
    h. Operation – shifter_operand:
        i. [11:8]: 0000, shift applied to Imm
        ii. [7:0]: 0000 0000, unsigned 8-bit immediate value, 3F in hexadecimal
d) What is the meaning of the instruction? : Copies the value of 0x00 into the eighth register.

Address 019 | Instruction E5898000

a) Change to binary format: 1110 0101 1000 1001 1000 0000 0000 0000
b) Write assembly code: STR $8, [$9, #0x000];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to pages A4-23, load register instruction specifies the destination register at bit[12:15] which is 1000 in binary and means Rd($8) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.

i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1001($9), and addr_mode is 0000 0000 0000.
ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
We have P == 1 and W == 0, the base register is not updated. (offset addressing)
The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($9) containing the base address and the immediate offset(#0x000) used with the value of Rn to form the address, that is [$9, #+000]
iii. Therefore, I can write the assembly code of this instruction like 'STR $8, [$9, #0x000];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 9th register with #0x000. Read the value saved in $8 and store it in the address [$9 + #0x000] of memory.

Address 01A | Instruction E5898004

a) Change to binary format: 1110 0101 1000 1001 1000 0000 0000 0100
b) Write assembly code: STR $8, [$9, #0x004];
c) Describe why you wrote the assembly code like the above: YOUR ANSWER
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to pages A4-23, load register instruction specifies the destination register at bit[12:15] which is 1000 in binary and means Rd($8) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1001($9), and addr_mode is 0000 0000 0100.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing)
        The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation

will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($9) containing the base address and the immediate offset(#0x004) used with the value of Rn to form the address, that is [$9, #+004]

iii. Therefore, I can write the assembly code of this instruction like 'STR $8, [$9, #0x004];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 9th register with #0x004. Read the value saved in $8 and store it in the address [$9 + #0x004] of memory.

Address 01B | Instruction E5898008

a) Change to binary format: 1110 0101 1000 1001 1000 0000 0000 1000
b) Write assembly code: STR $8, [$9, #0x008];
c) Describe why you wrote the assembly code like the above:
   a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
   b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
   c. Operation – Rd: According to pages A4-23, load register instruction specifies the destination register at bit[12:15] which is 1000 in binary and means Rd($8) for the loaded value.
   d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
      i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1001($9), and addr_mode is 0000 0000 1000.
      ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
      We have P == 1 and W == 0, the base register is not updated. (offset addressing)
      The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($9) containing the base address and the immediate offset(#0x008) used with the value of Rn to form the address, that is [$9, #+008]
      iii. Therefore, I can write the assembly code of this instruction like 'STR $8, [$9, #0x008];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 9th register with #0x008. Read the value saved in $8 and store it in the address [$9 + #0x008] of memory.

Address 01C | Instruction E589800C

a) Change to binary format: 1110 0101 1000 1001 1000 0000 0000 1100
b) Write assembly code: STR $8, [$9, #0x00C];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to pages A4-23, load register instruction specifies the destination register at bit[12:15] which is 1000 in binary and means Rd($8) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1001($9), and addr_mode is 0000 0000 1100.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing) The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($9) containing the base address and the immediate offset(#0x00C) used with the value of Rn to form the address, that is [$9, #+00C]
        iii. Therefore, I can write the assembly code of this instruction like 'STR $8, [$9, #0x00C];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.
d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 9th register with #0x00C. Read the value saved in $8 and store it in the address [$9 + #0x00C] of memory.

Address 01D | Instruction E5898010

a) Change to binary format: 1110 0101 1000 1001 1000 0000 0001 0000

b) Write assembly code: STR $8, [$9, #0x010];
c) Describe why you wrote the assembly code like the above: YOUR ANSWER
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to pages A4-23, load register instruction specifies the destination register at bit[12:15] which is 1000 in binary and means Rd($8) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1001($9), and addr_mode is 0000 0001 0000.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing)
        The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($9) containing the base address and the immediate offset(#0x010) used with the value of Rn to form the address, that is [$9, #+010]
        iii. Therefore, I can write the assembly code of this instruction like 'STR $8, [$9, #0x010];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.
d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 9th register with #0x010. Read the value saved in $8 and store it in the address [$9 + #0x010] of memory.

Address 01E | Instruction E5898014

a) Change to binary format: 1110 0101 1000 1001 1000 0000 0001 0100
b) Write assembly code: STR $8, [$9, #0x014];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check

the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)

c. Operation – Rd: According to pages A4-23, load register instruction specifies the destination register at bit[12:15] which is 1000 in binary and means Rd($8) for the loaded value.
d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
    i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1001($9), and addr_mode is 0000 0001 0100.
    ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
    We have P == 1 and W == 0, the base register is not updated. (offset addressing)
    The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($9) containing the base address and the immediate offset(#0x014) used with the value of Rn to form the address, that is [$9, #+014]
    iii. Therefore, I can write the assembly code of this instruction like 'STR $8, [$9, #0x014];' because the syntax of store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 9th register with #0x014. Read the value saved in $8 and store it in the address [$9 + #0x014] of memory.

Address 01F | Instruction E5898018

a) Change to binary format: 1110 0101 1000 1001 1000 0000 0001 1000
b) Write assembly code: STR $8, [$9, #0x018];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to pages A4-23, load register instruction specifies the destination register at bit[12:15] which is 1000 in binary and means Rd($8) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.

i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1001($9), and addr_mode is 0000 0001 1000.
ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
We have P == 1 and W == 0, the base register is not updated. (offset addressing)
The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 0, meaning that this operation will be a Store. Hence the operation has to be Store Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($9) containing the base address and the immediate offset(#0x018) used with the value of Rn to form the address, that is [$9, #+018]
iii. Therefore, I can write the assembly code of this instruction like 'STR $8, [$9, #0x018];' because the syntax of the store instruction is 'STR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 9th register with #0x018. Read the value saved in $8 and store it in the address [$9 + #0x018] of memory.

Address 020 | Instruction E59FDE78

a) Change to binary format: 1110 0101 1001 1111 1101 1110 0111 1000
b) Write assembly code: LDR $13, [$15, #0xE78];
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
    b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
    c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 1101 in binary and means Rd($13) for the loaded value.
    d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
        i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 1111, and addr_mode is 1110 0111 1000.
        ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
        We have P == 1 and W == 0, the base register is not updated. (offset addressing)
        The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 1, meaning that this operation

will be a Load. Hence the operation has to be Load Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($15) containing the base address and the immediate offset(#0xE78) used with the value of Rn to form the address, that is [$15, #+E78]

iii. Therefore, I can write the assembly code of this instruction like 'LDR $13, [$15, #0xE78];' because the syntax of load instruction is 'LDR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 15th register with #0xE78. Read the value saved in address [$15 + #0xE78] of memory and load it to $13.

Address 021 | Instruction E5931200

a) Change to binary format: 1110 0101 1001 0011 0001 0010 0000 0000
b) Write assembly code: LDR $1, [$3, #0x200];
c) Describe why you wrote the assembly code like the above:
   a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Load /store immediate offset' is only one instruction set encoding whose value at [25:27] bit is 010. So, I can figure out this instruction is a load/store register instruction.
   b. Operation – Condition Field: According to A4.1.23(Page A4-43), there is the detail of the Loed Register instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally. (Page A3-3)
   c. Operation – Rd: According to page A4-23, load register instruction specifies the destination register at bit[12:15] which is 0001 in binary and means Rd($1) for the loaded value.
   d. Operation – addressing mode: According to pages A4-43 in the ARM manual, the addressing mode is calculated like below.
      i. First, determine the I, P, U, W, Rn, and addr_mode bits of the instruction: we got the I bit is 0, the P bit is 1, the U bit is 1, the W bit is 0, Rn is 0011, and addr_mode is 0010 0000 0000.
      ii. Then determine the operation of addressing_mode: According to page A5-18 about Addressing Mode 2 - Load and Store Word or Unsigned Byte:
      We have P == 1 and W == 0, the base register is not updated. (offset addressing) The U bit is 1 indicates the offset is added to the base, and the B bit is 0 showing word access. Also, bit[20] representing the L bit is 1, meaning that this operation will be a Load. Hence the operation has to be Load Word Immediate Offset, and its syntax is [<Rn>, #+/-<offset_12>], where the register Rn($3) containing the base address and the immediate offset(#0x200) used with the value of Rn to form the address, that is [$3, #+200]
      iii. Therefore, I can write the assembly code of this instruction like 'LDR $1, [$3, #0x200];' because the syntax of load instruction is 'LDR{<cond>} <Rd>, <addr_mode>'.

d) What is the meaning of the instruction? : Calculate the address of memory by adding the value stored in the 3rd register with #200. Read the value saved in address [$3 + #0x200] of memory and load it to $1.

Address 022 | Instruction E3510001

a) Change to binary format: 1110 0011 0101 0001 0000 0000 0000 0001
b) Write assembly code: CMP $1 #0x01;
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Data processing immediate' is only one instruction set encoding whose values at [25:27] bit is 001 and the bit[20] can be 1 simultaneously. So, I can figure out this instruction is Data processing immediate instruction. Following the opcode, it is a CMP instruction. And this operation takes action to update flags after Rn - shift_operand.
    b. Operation – Condition Field: According to A3.4(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – I: I == 1, which means operand 2 is an immediate value
    d. Operation – Opcode: 1010 which means CMP - set condition codes on Op1 - Op2
    e. Operation – S: 1 = set condition codes, and it does not alter condition codes
    f. Operation – Rn: 0001, which is the 1st operand register
    g. Operation – SBZ: 0000, which is the destination register
    h. Operation – shifter_operand:
        i. Specifies the second operand. The options for this operand are described in Addressing Mode 1 - Data-processing operands on the page A5-2, including how each option causes the I bit (bit[25]) and the shifter_operand bits (bits[11:0]) to be set in the instruction. So that we can double check the operation is CMP.
        ii. According to page A5-3, I == 1 so that the operation should be 32-bit immediate, the rotate_imm is 0000, that is shift applied to Imm, and bits[0:7] 0000 0001 is an Unsigned 8-bit immediate value.
        iii. Specifies the immediate constant wanted. It is encoded in the instruction as an 8-bit immediate (immed_8) and a 4-bit immediate (rotate_imm), so that <immediate> is equal to the result of rotating immed_8 right by (2 × rotate_imm) bits which is still 0000 0001
        iv. According to the page A5-6, rotate_imm == 0, so that "shift_carry_out = C flag"
d) What is the meaning of the instruction? : If we subtract #0x01 from the value of register $1, we get the alu_out which is the base of updating the condition flags: Set Z if alu_out == 0, that is the value of $1 equals #0x01, and N Flag same as alu_out[31], set C if there is not borrowfrom(Rn - shifter_operand), set V if there is an overflow from (Rn - shifter_operand)

Address 023 | Instruction 0A000000

a) Change to binary format: 0000 1010 0000 0000 0000 0000 0000 0000
b) Write assembly code: BEQ #37
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Branch and branch with link' is only one instruction set encoding whose value at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
    b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 0000 and it means the instruction Branch (B in assembly language) becomes BEQ for "Branch if Equal", which means the Branch will only be taken if the Z flag is set.
    c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn't need to store any return address because the L bit is 0.
    d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
        i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 00 0000 0000 0000 0000 0000 0000 0000. (Because the signed immediate is 0000 0000 0000 0000 0000 0000 here.)
        ii. Then, get 0000 0000 0000 0000 0000 0000 0000 0000 by shifting the result left two bits.
        iii. Because the address of this instruction is 35, the content of the PC will be 35*4 + 8 bytes. So, the target address will be (35*4+8) + 0 = 148(bytes). It means after the operation of this instruction, the PC will be moved to 148/4 = 37.
        iv. Therefore, I can write the assembly code of this instruction like 'BEQ #37;' because the syntax of the branch instruction is 'B{L}{cond} <target_address>'.
d) What is the meaning of the instruction? : If the Z value of the flag register is set then move to address 037, otherwise (the Z value of the flag register is not set) move to address 024.

Address 024 | Instruction EAFFFFFB

a) Change to binary format: 1110 1010 1111 1111 1111 1111 1111 1011
b) Write assembly code: B #33
c) Describe why you wrote the assembly code like the above:
    a. Type of instruction: According to the figure A3-1 in the ARM manual, 'Branch and branch with link' is only one instruction set encoding whose values at [25:27] bit is 101. So, I can figure out this instruction is a branch instruction.
    b. Operation – Condition Field: According to A4.1.5(Page A4-10), there is the detail of the branch instruction. The 'Operation' part of the instruction said that I should check the condition is passed first. The condition field of this instruction is 1110 and it means the instruction can operate unconditionally.
    c. Operation – L: According to pages A4-10, branch instruction branches without storing a return address when L is omitted. In the case of this instruction, it doesn't need to store any return address because the L bit is 0.

d. Operation – Target Address: According to pages A4-10 in the ARM manual, the target address is calculated like below.
   i. First, the result of sign-extending the 24-bit signed immediate to 30 bits is 11 1111 1111 1111 1111 1111 1111 1011. (Because the signed immediate is 1111 1111 1111 1111 1111 1111 1011 here.)
   ii. Then, get 1111 1111 1111 1111 1111 1111 1110 1100 by shifting the result left two bits.
   iii. Because the address of this instruction is 36, the content of the PC will be 4*36 + 8 bytes. So, the target address will be (4*36+8) - 20 = 132(bytes). It means after the operation of this instruction, the PC will be moved to 132/4 = 33.
   iv. Therefore, I can write the assembly code of this instruction like 'B #33;' because the syntax of the branch instruction is 'B{L}{cond} <target_address>'.

d) What is the meaning of the instruction? : The instruction means 'branch to address 33'.

Explain the actual execution flow of the instructions(Address 000~024)

| | | | |
|---|---|---|---|
| 1 | Address 000 | B #8; | Branch to address 8 |
| 2 | Address 008 | LDR $2, [$15, #0xEC8]; | Read the value saved in the address [$15 + #0xEC8] of memory and load it to the second register $2. |
| 3 | Address 009 | MOV $0 #0x40; | Copies the value of 0x40 into the zeroth register. |
| 4 | Address 00A | STR $0, [$2, #0x010]; | Read the value saved in $0 and store it in the address [$2 + #0x010] of memory. |
| 5 | Address 00B | STR $0, [$2, #0x014]; | Read the value saved in $0 and store it in the address [$2 + #0x014] of memory. |
| 6 | Address 00C | STR $0, [$2, #0x018]; | Read the value saved in $0 and store it in the address [$2 + #0x018] of memory. |
| 7 | Address 00D | STR $0, [$2, #0x01C]; | Read the value saved in $0 and store it in the address [$2 + #0x01C] of memory. |
| 8 | Address 00E | STR $0, [$2, #0x020]; | Read the value saved in $0 and store it in the address [$2 + #0x020] of memory. |
| 9 | Address 00F | STR $0, [$2, #0x024]; | Read the value saved in $0 and store it in the address [$2 + #0x024] of memory. |
| 10 | Address 010 | MOV $0 #0x3F; | Copies the value of 0x3F into the zeroth register. |

| | | | |
|---|---|---|---|
| 11 | Address 011 | STR $0, [$2, #0x028]; | Read the value saved in $0 and store it in the address [$2 + #0x028] of memory. |
| 12 | Address 012 | MOV $0 #0x08; | Copies the value of 0x08 into the zeroth register. |
| 13 | Address 013 | STR $0, [$2, #0x02C]; | Read the value saved in $0 and store it in the address [$2 + #0x02C] of memory. |
| 14 | Address 014 | LDR $3, [$15, #0xE9C]; | Read the value saved in the memory address [$15 + #0xE9C] and load it to the third register $3. |
| 15 | Address 015 | LDR $1, [$15, #0xE9C]; | Read the value saved in the memory address [$15 + #0xE9C] and load it to the first register $1. |
| 16 | Address 016 | STR $1, [$3, #0x000]; | Read the value saved in $1 and store it in the address [$3 + #0x000] of memory. |
| 17 | Address 017 | LDR $9, [$15, #0xE98]; | Read the value saved in the memory address [$15 + #0xE98] and load it to the ninth register $9. |
| 18 | Address 018 | MOV $8 #0x00; | Copies the value of 0x00 into the eighth register. |
| 19 | Address 019 | STR $8, [$9, #0x000]; | Read the value saved in $8 and store it in the address [$9 + #0x000] of memory. |
| 20 | Address 01A | STR $8, [$9, #0x004]; | Read the value saved in $8 and store it in the address [$9 + #0x004] of memory. |
| 21 | Address 01B | STR $8, [$9, #0x008]; | Read the value saved in $8 and store it in the address [$9 + #0x008] of memory. |
| 22 | Address 01C | STR $8, [$9, #0x00C]; | Read the value saved in $8 and store it in the address [$9 + #0x00C] of memory. |
| 23 | Address 01D | STR $8, [$9, #0x010]; | Read the value saved in $8 and store it in the address [$9 + #0x010] of memory. |
| 24 | Address 01E | STR $8, [$9, #0x014]; | Read the value saved in $8 and store it in the address [$9 + #0x014] of memory. |
| 25 | Address 01F | STR $8, [$9, #0x018]; | Read the value saved in $8 and store it in the address [$9 + #0x018] of memory. |
| 26 | Address 020 | LDR $13, [$15, #0xE78]; | Read the value saved in address [$15 + |

| | | | |
|---|---|---|---|
| | | | #0xE78] of memory and load it to $13. |
| 27 | Address 021 | LDR $1, [$3, #0x200]; | Read the value saved in address [$3 + #0x200] of memory and load it to $1. |
| 28 | Address 022 | CMP $1 #0x01; | Set Z if the value of register $1 is equal to the value of #0x01. |
| 29 | Address 023 | BEQ #37 | If the Z value of the flag register is set, move to address 037 |
| | | | If the Z value of the flag register is not set, move to address 024 |
| 30 | Address 024 | B #33 | Branch to address 33 |

Specify where the execution ends (If not, specify the range repeated in detail)

Before the instruction of 022, there is only one possibility, however, after we did the compare operation, the condition flags could be updated. If Z set, which means we have the value in $1 sames as #0x01, the BEQ(branch if equal) would be done. After “BEQ #37” has been executed, it generates two ways to go: If it belongs to Case a in instruction 023(Z set), it moves to instruction 037, but instruction starting at address 025 is not analyzed in this report and therefore not discussed. If it belongs to Case b(Z clear) in the instruction at address 023, it moves to address 024 and then branches to instruction 033. Similarly, instruction 033 is outside the scope of our instruction list, and we cannot predict the next behavior of the pc.