(15) **Question 1.** Select A, B, C, D, or E that best answers the question. Put answers in the boxes.

What type of computer is the ARM Cortex M?

A) Von Neumann
B) Big endian
C) **Harvard** (separate bus for fetching op code from fetching data)
D) CISC
E) None of the above

Which of the following is a rule for proper use of the stack?

A) **To pop, we first read at SP then increment the SP by 4**
B) To push, first write at SP then decrement the SP by 4
C) It is ok to access memory locations above the SP (e.g., SP-8)
D) The software must execute something like LDR SP,=InitialSP to initialize the stack pointer
E) None of the above

Why does the Cortex M have multiple buses?

A) To reduce the number of mistakes software can make
B) To allow the microcontroller to access I/O ports
C) **To speed up execution by allowing multiple actions to run in parallel**
D) To reduce power saving energy
E) None of the above

What is a data flow graph?

A) A drawing with circles and rectangles. The circles are software modules and the rectangles are hardware. If module A invokes an operation in module B there is an arrow from A to B.
B) A drawing that describes the sequence of operations of software, defining what and when software actions will occur.
C) To describe how data are stored in a computer, we draw a picture or graph of how the data are organized
D) **A drawing with circles and rectangles. The circles are software modules and the rectangles are hardware. If a module A passes data to module B, then an arrow is drawn from A to B.**
E) None of the above

What is open collector logic?

A) It is logic used when the microcontroller creates an input port.
B) Logic that has two states, high and low.
C) **Logic that has two states, low and off.**
D) Logic that has three states, high, low, and off.
E) None of the above
(15) Question 2. Interface the LED to PG0 such that if PG0 is low, the LED is on, and if PG0 is high
the LED is off. The desired LED operating point is 1.0V at 2 mA. The $V_{OH}$ of the microcontroller is
3.1 V. The $V_{OL}$ of the microcontroller is 0.3 V. The $V_{OL}$ of the 7406 is 0.5 V. Your bag of parts
includes the switch, the 7406, the LED, and resistors (you specify the values). Pick the fewest
components to use. You will not need them all. You may also use 3.3V, 5V power and ground. Show
the equations used to calculate the resistor value.

$$R = \frac{(3.3-1-0.3V)}{2mA} = \frac{2V}{2mA} = 1000\Omega$$
(10) Question 3. Write an assembly subroutine, called \texttt{Mul7}, that multiplies by 7 using just shifts and adds (not the MUL instruction). The input is passed by value in Register R0, and the output is returned in Register R0. You may use Registers R1, R2, R3, R12 as scratch registers without saving and restoring them. Notice that if $x$ is a variable, then $7x = 8x - x$, and also $7x = 4x + 2x + x$.

\begin{verbatim}
 Mul7 RSB R0, R0, R0, LSL #3 ;8*in-in
 BX LR
 Mul7 LSL R1, R0, #3 ;8*in
 SUB R0, R0, R1 ;in*7
 BX LR
 Mul7 MOV R1, R0 ;in
 LSL R0, #1 ;2*in
 ADD R1, R1, R0 ;in*3
 LSL R0, #1 ;4*in
 ADD R0, R1, R0 ;in*7
 BX LR
\end{verbatim}

(15) Question 4. State the term that is best described by each definition.

**Part a)** A property of ROM such that data is not lost if power is removed and then restored.

- **nonvolatile**

**Part b)** The error that occurs after an addition, subtraction, or multiplication such that the result does not fit back into the register.

- **overflow**

**Part c)** A quality of a software system defined by how much memory is required.

- **Static efficiency**

**Part d)** The name given to describe 1,048,576 ($2^{20}$) bytes.

- **mebibyte**

**Part e)** A type of digital logic where the voltage representing false is less than the voltage representing true.

- **positive logic**

(30) Question 5. Assume that Port G is already initialized such that PG0 is an output and PG2 is an input. You do not have to write the initialization code. Write C function that toggles PG0 16 times.
However, before each time your function outputs to PG0, it should read the PG2 input. If PG2 is low, your function stops toggling and waits until PG2 becomes high. After PG0 has been toggled 16 times, your function should return. You may define additional variables. Bit-specific addressing is allowed but not required. You may use the following definition

```
#define GPIO_PORTG_DATA_R (*((volatile unsigned long *)0x400263FC))
```

This figure illustrates one possible case. PG0 toggles only while PG2 is high.

![Diagram of PG2 and PG0 inputs and outputs]

In this second possible case, the PG2 input is always high. In this case your function toggles PG2 16 times and then returns.

![Diagram of PG2 and PG0 inputs and outputs with PG2 high]

In this third possible case, the PG2 input remains low forever. For this case, your function never toggles PG0 and never returns.

```c
void PortG_Toggle(void){ long i;
    for(i=0; i<16; i++){
        while((GPIO_PORTG_DATA_R&0x04)==0){}; // wait for PG2 high
        GPIO_PORTG_DATA_R ^= 0x01;  // toggle PG0
    }
}
```

```c
void PortG_Toggle(void){ long count=16;
    while(count){
        while((GPIO_PORTG_DATA_R&0x04)==0){}; // wait for PG2 high
        GPIO_PORTG_DATA_R ^= 0x01;  // toggle PG0
        count--;
    }
}
```

```c
#define PG2 (*((volatile unsigned long *)0x40026010))
#define PG0 (*((volatile unsigned long *)0x40026004))
void PortG_Toggle(void){ long i;
    for(i=0; i<16; i++){
        while(PG2 == 0){}; // wait for PG2 high
        PG0 ^= 0x01;       // toggle PG0
    }
}
```
(15) Question 6. Consider the following assembly language system. The subroutine Circ calculates the circumference of a circle. The input is passed in using R0 in cm, and the output is returned in R0, also in cm. \( 2\pi \) is approximated by \( 6283185/1000000 \). The assembly listing is shown.

```
0x000006CC          Circ
0x000006CC B502               PUSH  {R1,LR}
0x000006CE 49FA               LDR   R1,=6283185
0x000006D0 F000F801           BL    Multiply
0x000006D4 BD02               POP   {R1,PC}
0x000006D6          Multiply
0x000006D6 FB00F001           MUL   R0,R0,R1
0x000006DA 49F8               LDR   R1,=1000000
0x000006DC FBB0F0F1           UDIV  R0,R0,R1 ;return R0*R1/1000000
0x000006E0 4770               BX    LR
0x000006E2          Start
0x000006E2 F04F007B           MOV   R0,#123  ;radius in cm
0x000006E6 F7FFFFF1           BL    Circ     ;returns 772 cm in R0
0x000006EA E7FE     Loop      B     Loop
```

Part a) In the first line of Circ, what does 0x000006CC represent? -----------------

Part b) In the first line of Circ, what does B502 represent? --------------------------

Part c) What value is in the link register immediately after the BL Multiply instruction is executed?  ------------------------------------------------------

Part d) What addressing mode does the LDR R1,=6283185 instruction use?--

Part e) Assume the stack is initially empty at the beginning of Start. In particular the SP has the value 0x20001000. Assume initial value of R1 is 0 at the beginning of Start. During the execution of the Multiply subroutine, draw a stack picture illustrating all values on the stack, and the exact location of the SP at this time.

```
<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x000006CC</td>
<td>Circ</td>
</tr>
<tr>
<td>0x000006CC</td>
<td>B502</td>
</tr>
<tr>
<td>0x000006CE</td>
<td>49FA</td>
</tr>
<tr>
<td>0x000006D0</td>
<td>F000F801</td>
</tr>
<tr>
<td>0x000006D4</td>
<td>BD02</td>
</tr>
<tr>
<td>0x000006D6</td>
<td>Multiply</td>
</tr>
<tr>
<td>0x000006D6</td>
<td>FB00F001</td>
</tr>
<tr>
<td>0x000006DA</td>
<td>49F8</td>
</tr>
<tr>
<td>0x000006DC</td>
<td>FBB0F0F1</td>
</tr>
<tr>
<td>0x000006E0</td>
<td>4770</td>
</tr>
<tr>
<td>0x000006E2</td>
<td>Start</td>
</tr>
<tr>
<td>0x000006E2</td>
<td>F04F007B</td>
</tr>
<tr>
<td>0x000006E6</td>
<td>F7FFFFF1</td>
</tr>
<tr>
<td>0x000006EA</td>
<td>E7FE</td>
</tr>
</tbody>
</table>
```

The address of the instruction in ROM

Object code of the instruction

0x000006D4* (actually 0x00006D5)

PC relative

* actually the saved value of the return address will be 0x000006EB (so the T bit remains set)