Second Semester Examination
2016/2017 Academic Session

June 2017

CST432 – Microprocessors & Embedded Systems

[Mikropemproses & Sistem Terbenam]

Duration : 2 hours
[Masa : 2 jam]

INSTRUCTIONS TO CANDIDATE:
[ARAHAN KEPADA CALON:]

- Please ensure that this examination paper contains **FOUR** questions in **SEVEN** printed pages before you begin the examination.

  *[Sila pastikan bahawa kertas peperiksaan ini mengandungi **EMPAT** soalan di dalam **TUJUH** muka surat yang bercetak sebelum anda memulakan peperiksaan ini.]*

- Answer **ALL** questions.

  *[Jawab **SEMUA** soalan.]*

- You may answer the questions either in English or in Bahasa Malaysia.

  [Anda dibenarkan menjawab soalan sama ada dalam bahasa Inggeris atau bahasa Malaysia.]

- In the event of any discrepancies, the English version shall be used.

  [Sekiranya terdapat sebarang percanggahan pada soalan peperiksaan, versi bahasa Inggeris hendaklah diguna pakai.]

1. (a) Microcontrollers are generally used in Embedded Systems design due to their unique characteristics.

Mikropengawal biasanya digunakan dalam reka bentuk Sistem Terbenam disebabkan sifat unik mereka.

(i) State **two** (**2**) additional hardware components found in microcontrollers compared with microprocessors using the same instruction set architecture.

*Nyatakan **dua** (**2**) komponen perkakasan tambahan yang terdapat dalam mikropengawal berbanding dengan mikropemproses yang menggunakan seni bina set suruhan serupa.*

(4/100)

(ii) Microcontrollers are often found in battery-operated devices, state **three** (**3**) characteristics of microcontrollers that enable them to function effectively in such systems.

*Mikropengawal biasanya terdapat dalam peranti beroperasi bateri, nyatakan **tiga** (**3**) ciri mikropengawal yang membolehkan mereka berfungsi secara berkesan dalam sistem tersebut.*

(6/100)

(b) Microcontroller architectures often provide co-processor support in the instruction set.

Seni bina mikropengawal biasanya membekalkan sokongan untuk ko-pemproses dalam set suruhannya.

(i) State **two** (**2**) examples of co-processors commonly found in microcontrollers today.

*Nyatakan **dua** (**2**) contoh ko-pemproses yang sering dijumpai dalam mikropengawal terkini.*

(4/100)

(ii) A new microcontroller is being designed to support live video streaming using the x.265 video compression/decompression (codec) standard, state **three** (**3**) different hardware approaches for implementing a x.265 codec using co-processors.

*Suatu mikropengawal baru akan direka bentuk untuk menampung strim video langsung secara menggunakan piawaian pengekodan/ penyahkodan (codec) video x.265, nyatakan **tiga** (**3**) kaedah perkakasan berbeza untuk mengimplementasikan codec x.265 secara ko-pemproses.*

(6/100)

(iii) Which of the approaches in 1(b)(ii) will be able to support a new streaming video codec that is still under development? Justify your answer.

Kaedah daripada 1(b)(ii) manakah mampu menampung codec strim video baru yang masih dalam pembangunan? Beri sokongan untuk jawapan anda.

(5/100)

2. ARMv4T processors support ARM and Thumb states with different instruction sizes.

Mikropemproses ARMv4T menyokong keadaan-keadaan ARM dan Thumb yang menggunakan saiz suruhan berbeza.

(a) State **one** (**1**) advantage and **one** (**1**) disadvantage of using Thumb state in the ARMv4T architecture.

*Nyatakan **satu** (**1**) kelebihan dan **satu** (**1**) kekurangan menggunakan keadaan Thumb dalam seni bina ARMv4T.*

(4/100)

(b) (i) How does the ARMv4T-based processor determine which state the processor is currently in?

Bagaimanakah pemproses berasaskan ARMv4T mengenal pasti keadaan pemproses terkini?

(ii) The Branch and Exchange (BX) Instruction is used to switch states. How does the processor know what is the target (new) state for the next instruction?

Suruhan Cabang dan Tukar (BX) digunakan untuk tukar keadaan. Bagaimanakah pemproses kenal pasti keadaan sasaran (baru) untuk suruhan berikutnya?

(iii) Interworking is the technique used to support calling ARM subroutines from Thumb state and vice-versa. What is the penalty for calling many **different** ARM subroutines directly from Thumb state?

*Teknik "Interworking" digunakan untuk membolehkan panggilan subrutin ARM daripada keadaanThumb dan sebaliknya. Apakah penalti memanggil banyak subrutin ARM yang **berbeza** secara terus daripada keadaan Thumb?*

(9/100)

(c) A subroutine Sub1() compliant with the **Procedure Call Standard for the ARM Architecture** (AAPCS) requires five (5) parameters: P1, P2, P3, P4, P5.

*Suatu subrutin Sub1() yang mematuhi **Piawaian Panggilan Prosedur seni bina ARM** (AAPCS) memerlukan lima (5) parameter: P1, P2, P3, P4, P5.*

(i) Explain how each of these five parameters are passed into Sub1() from the main program.

Jelaskan bagaimana setiap lima parameter tersebut dihulurkan kepada Sub1() daripada atur cara utama.

(ii) Given that Sub1() does not call any other subroutines, is it necessary to preserve the contents of register R14 before invoking Sub1() from the main program? Justify your reasons.

Diberi Sub1() tidak memanggil subrutin-subrutin lain, adakah ia mustahak untuk memelihara kandungan daftar R14 sebelum melaksanakan Sub1() daripada atur cara utama? Berikan justifikasi anda.

(12/100)

3. Given that R0=0x001F3B90, R1=0x04, R2=0x08, and R3=0xF180A912 ***at the start of each of the following sections***, determine the outcome of the execution of the following 32-bit ARM instructions. The intermediate calculation steps must be shown in your answers. Provide the answer in the following format: name of register or address for the result of the final executed instruction, and the value of its contents. Note: all values are given as hexadecimal numbers.

Diberi R0=0x001F3B90, R1=0x04, R2=0x08, dan R3=0xF180A912 ***pada permulaan setiap bahagian tersebut****. Apakah hasil selepas perlaksanaan jujukan suruhan ARM 32-bit berikut? Langkah pengiraan haruslah diberi bersama jawapan anda. Berikan jawapan secara: nama daftar ataupun alamat ingatan untuk hasil suruhan terakhir dan nilai kandungannya. Nota: semua nilai adalah dalam nombor "hexadecimal".*

(a)
```
ADDS    R0, R3, R0
MOVEQ   R0, #0xF0
```

Register = _____ Value = _____

Daftar = _____ *Nilai* = _____

(4/100)

(b)
```
ORR     R4, R3, R0
ASL     R4, R4, R2
```

Register = _____ Value = _____

Daftar = _____ *Nilai* = _____

(4/100)

```
        SUB     R0, R0, #1
LOOP:   STRB    R3, [R0, R1]
        ROR     R3, R3, R2
        SUBS    R1, R1, #1
        BNE     LOOP
```

Memory Address1 = _____ Value1 = _____
Memory Address2 = _____ Value2 = _____
Memory Address3 = _____ Value3 = _____
Memory Address4 = _____ Value4 = _____

Alamat Ingatan1 = _____ *Nilai1* = _____
Alamat Ingatan2 = _____ *Nilai2* = _____
Alamat Ingatan3 = _____ *Nilai3* = _____
Alamat Ingatan4 = _____ *Nilai4* = _____

(17/100)

4. A LEGO MINDSTORMS-based meal ordering system is used to manage orders at a cafeteria. There are three token dispensers each controlled by a separate MINDSTORMS motor. Customers press the three top buttons, *Left*, *Square*, or *Right*, on the MINDSTORMS NXT for different set meals, A, B, or C. After the button press, the meal selection is *displayed* on the LCD, and one of the MINDSTORMS motors is used to release the correct meal order token. You are required to write a program in ***ARM Assembly Language*** for the meal ordering system.

   You do not need to worry about system configuration and other processor hardware related details. However, the button press on the NXT is returned as a raw A/D Converter value by `get_button_adc_value()`. There should be a delay of 50 ms between consecutive reads of the A/D values. This raw A/D value will need to be debounced since the value does not settle to a stable value immediately. Three consecutive detection of a particular button-press value is needed to confirm the actual button press.

   Control of the token dispenser is accomplished by specifying the respective motor value when calling the provided subroutine `dispense_token()`. Only the data declaration section and the main program (main) containing ARM Assembly code for the meal ticketing system need to be shown in your answer.

   Suatu sistem pesanan makanan berdasarkan LEGO MINDSTORMS digunakan untuk melancarkan pesanan sesuatu kafeteria. Terdapat tiga pengagih token yang dikawal oleh motor MINDSTORMS berasingan. Pelanggan tekan sesuatu butang atasan, Kiri, Segi-Empat atau Kanan pada MINDSTORMS NXT untuk memilih hidangan set A, B, atau C. Setelah butang ditekan, pilihan hidangan dipaparkan pada skrin LCD, dan suatu motor MINDSTORMS digunakan untuk mengagih token pesanan hidangan yang betul. Anda dikehendaki menulis atur cara dalam ***Bahasa Himpunan ARM*** *untuk sistem pesanan makanan tersebut.*

Anda tidak perlu mengambil kira konfigurasi sistem ataupun konfigurasi perkakasan terperinci yang lain. Namun, tekanan butang pada NXT dipulangkan secara nilai Pengantar A/D mentah oleh `get_button_adc_value()`*. Masa lengah (delay) 50 ms harus diamalkan di antara bacaan nilai A/D berturut-turut. Nilai A/D tersebut perlu dinyah-lantun (debounce) sebab nilai tersebut tidak menjadi stabil secara serta-merta. Tiga bacaan nilai tekanan butang tertentu diperlukan untuk mengenalpasti tekanan butang yang sebenar.*

Kawalan pengagih token dilakukan secara menentukan nilai motor berkenaan semasa memanggil subrutin `dispense_token()` *yang dibekal. Hanya kod Himpunan ARM untuk bahagian pengisytiharan data serta atur cara utama (main) untuk sistem pesanan makanan perlu ditunjukkan dalam jawapan anda.*

The following definitions and subroutines (in C language) are provided for your use:

Takrifan dan subrutin (dalam Bahasa C) berikut dibekalkan untuk kegunaan anda:

```
#define MEAL_A_MOTOR_PORT 0
#define MEAL_B_MOTOR_PORT 1
#define MEAL_C_MOTOR_PORT 2

/** Raw 10-bit A/D Converter Threshold Values for Button Presses
 *
 * They are given in sequence from high to low thresholds.
 * If the value exceeds the given A/D threshold value
 * it is considered as a button press.
 * e.g., if Raw A/D Value = 350 it is a Right Button Press (>256)
 *       if Raw A/D Value = 20 then no button is pressed (<64)
 **/
#define SQUARE_BUTTON_THRESH 1535
#define RIGHT_BUTTON_THRESH   256
#define LEFT_BUTTON_THRESH     64

#define SYSTICK_50MS     50
#define SYSTICK_1000MS 1000

typedef unsigned long U32;
typedef unsigned char U8;

meal_a_str: .asciz "Meal A"
meal_b_str: .asciz "Meal B"
meal_c_str: .asciz "Meal C"


/** Sleep for @a ms milliseconds.
 *
 * @param ms The number of milliseconds to sleep.
 *
 */
void nx_systick_wait_ms(U32 ms);

/** Enable @a sensor in analog mode.
 *
 * @param sensor The sensor port.
*/
```

```
/**
 * Display Content string on LCD (Row 2)
 *    @param string: Null-terminated string
 */
void nx_progcontent(char *string);

/** Retrieve the Raw 10-bit A/D Converter Value.
 *
 * @return The Raw 10-bit A/D Converter Value in a 32-bit register.
 */
U32 get_button_adc_value();

/** Dispense a token for the @a set meal.
 *
 * @param motor The token dispenser motor port.
 */
void dispense_token(U32 motor);
```

(25/100)

- oooOooo -