UNIVERSITI SAINS MALAYSIA

Second Semester Examination
2015/2016 Academic Session

June 2016

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) Microprocessor use on-chip registers to store temporary values and perform calculations.

Mikropemproses menggunakan daftar dalam cip untuk menyimpan nilai sementara dan melakukan pengiraan.

(i) State **four** (**4**) type of register file designs that have been implemented in microprocessors.

*Nyatakan **empat** (**4**) jenis reka bentuk fail daftar yang telah dilaksanakan dalam mikropemproses.*

(8/100)

(ii) Given that instruction and data caches are used to speed up access to main memory in a computer system, justify which of the four register file designs in 1(a)(i) would be the ***least negatively affected*** by a small data cache size.

*Diberi cache data dan suruhan digunakan untuk mempercepatkan capaian kepada ingatan utama sesuatu sistem komputer, justifikasikan yang manakah antara empat reka bentuk fail daftar daripada 1(a)(i) akan **paling kurang terjejas** oleh saiz cache data yang kecil.*

(4/100)

(b) Microprocessors need to access peripheral devices to perform various I/O tasks.

Mikropemproses perlu mencapai peranti persisian untuk melakukan pelbagai tugas I/O.

(i) State **two** (**2**) I/O architectures that have been adopted in microprocessor designs to interconnect peripheral devices to the microprocessor.

*Nyatakan **dua** (**2**) seni bina I/O yang diamalkan dalam reka bentuk mikropemproses untuk menyaling sambung peranti persisian kepada mikropemproses.*

(4/100)

(ii) Given that computer systems have largely standardized on one of the two I/O architectures given in 1(b)(i), briefly explain **three** (**3**) advantages of the I/O architecture chosen compared to the alternative.

*Diberi kebanyakan sistem komputer telah memilih suatu seni bina I/O daripada 1(b)(i) sebagai piawaian, jelaskan secara ringkas **tiga** (**3**) kelebihan seni bina I/O yang dipilih berbanding dengan alternatifnya.*

(9/100)

2. Stacks are used to support subroutine calls in microprocessors.

Tindanan digunakan untuk menyokong panggilan subrutin pada mikropemproses.

(a) The ARMv4T architecture does not have a dedicated stack pointer. Briefly explain how the typical functions of PUSH and POP for a stack is implemented in the ARMv4T architecture using available ARMv4T instructions and registers.

Seni bina ARMv4T tidak mempunyai penuding tindanan khusus. Jelaskan secara ringkas bagaimana fungsi biasa PUSH dan POP sesuatu tindanan diamalkan dalam seni bina ARMv4T menggunakan suruhan dan daftar ARMv4T yang sedia ada.

(8/100)

(b) Given that a C subroutine with prototype `void sub1(int paramA, int paramB)`; which accepts two 32-bit integer parameters `paramA` and `paramB` needs to be called from an ARM assembly routine, briefly explain how the two parameters will be passed to `sub1()` according to the ARM Architecture Procedure Call Standard (AAPCS).

Diberi suatu subrutin C dengan prototaip `void sub1(int paramA, int paramB)`; *yang menerima dua parameter integer 32-bit* `paramA` *dan* `paramB` *perlu dipanggil daripada rutin himpunan ARM, jelaskan secara ringkas bagaimana kedua-dua parameter tersebut akan dihantar kepada* `sub1()` *mengikut Piawaian Panggilan Subrutin Seni Bina ARM (AAPCS).*

(4/100)

(c) Given that `sub2()` is a leaf (stub) routine which uses registers R0, R1, R3, R4, R7, and R8 for implementing its algorithm, state the minimum list of registers that **MUST** be pushed to the stack by `sub2()` according to the AAPCS in order for the calling routine to work correctly after returning from `sub2()`.

Diberi `sub2()` *adalah rutin dedaun (tunggul) yang menggunakan daftar R0, R1, R3, R4, R7, dan R8 untuk melaksanakan algoritmanya, nyatakan senarai daftar minimum mengikut AAPCS yang **MESTI** ditolak ke dalam tindanan oleh* `sub2()`*, supaya rutin yang dipanggilnya dapat berfungsi secara betul selepas pulang daripada* `sub2()`*.*

(6/100)

(d) Most programming languages support only simple types such as a single integer value as the return value from subroutines. Briefly explain why contents of complex structures are not easily returned by subroutines, based on available parameter passing and return techniques.

Kebanyakan bahasa pengaturcaraan hanya menyokong jenis mudah seperti nilai integer tunggal sebagai nilai pulangan subrutin. Jelaskan secara ringkas mengapakah kandungan struktur kompleks tidak mudah dipulangkan oleh subrutin, berdasarkan teknik huluran dan pulangan parameter yang sedia ada.

(7/100)

3. Given that R4=0x3011F9C7, R5=0x0103, R6=0x02, and R7=0x10A5BC3F ***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 R4=0x3011F9C7, R5=0x0103, R6=0x02, dan R7=0x10A5BC3F ***pada permulaan setiap bahagian tersebut****, tentukan hasil selepas perlaksanaan jujukan suruhan ARM 32-bit berikut. Langkah pengiraan haruslah diberi bersama jawapan anda. Berikan jawapan mengikut format berikut: nama daftar ataupun alamat ingatan untuk hasil suruhan terakhir dan nilai kandungannya. Nota: semua nilai adalah dalam nombor "hexadecimal".*

(a)
```
ORR   R1, R7, R4
ADD   R1, R5, R1
```

Register = _____ Value = _____

Daftar = _____ *Nilai* = _____

(4/100)

(b)
```
SUB   R3, R7, R5 ASL R6
AND   R9, R4, R3
```

Register = _____ Value = _____

Daftar = _____ *Nilai* = _____

(4/100)

(c)
```
      MOV    R2, #2
LOOP: STRB   R7, [R4], #2
      ASR    R7, R7, R6
      SUBS   R2, R2, #1
      BNE    LOOP
```

Memory Address1 = _____ Value1 = _____
Memory Address2 = _____ Value2 = _____

Alamat Ingatan1 = _____ *Nilai1* = _____
Alamat Ingatan2 = _____ *Nilai2* = _____

(13/100)

(d) STRH R7, [R4, R5]

Memory Address1 = _____ Value1 = _____
Memory Address2 = _____ Value2 = _____

Alamat Ingatan1 = _____ *Nilai1* = _____
Alamat Ingatan2 = _____ *Nilai2* = _____

(4/100)

4. A LEGO MINDSTORMS-based aquarium cleaning system is used to maintain the cleanliness of an aquarium by checking the amount of suspended particles (turbidity) in the water. The Light Sensor shines the LED light through a transparent glass tube with flowing aquarium water for sampling. Dirty water will have a higher amount of reflected light compared with clean water. Hence a ***high*** light sensor reading (less reflection) indicates clean water, while a ***low*** light sensor reading (higher reflection) indicates dirty water. The constant `DIRTY_THRESHOLD` will be used to activate the aquarium filter pump, which will activate until the light sensor readings increase above `DIRTY_THRESHOLD` for 60 seconds. You are required to write a program in ***ARM Assembly Language*** for the aquarium cleaning system.

You do not need to worry about system configuration and other processor hardware related details. Control of the filter pump is done using provided subroutines `start_filter_pump()` and `stop_filter_pump()`. Only the data declaration section and the main program (main) containing ARM Assembly code for the aquarium cleaning system need to be shown in your answer.

Note: The Light Sensor and Filter Pump should be initialized using the correct port number (defined as `LIGHT_PORT` and `PUMP_MOTOR_PORT`) at the beginning of the program. In addition, the Light Sensor LED light should be enabled using `nx_sensors_analog_digi_set()`.

*Suatu sistem pembersihan akuarium berdasarkan LEGO MINDSTORMS digunakan untuk menjaga kebersihan suatu akuarium secara mengesan jumlah partikel tergantung dalam air. Penderia Cahaya menyinarkan cahaya LED melalui tiub gelas lutsinar yang mengandungi air akuarium untuk mendapat sampel. Air kotor akan mengandungi jumlah cahaya bayangan yang lebih tinggi berbanding dengan air jernih. Maka, bacaan penderia cahaya yang **tinggi** (kurang bayangan) menunjukkan airnya jernih, manakala bacaan penderia cahaya yang **rendah** (lebih bayangan) menunjukkan airnya kotor. Nilai konstan `DIRTY_THRESHOLD` akan digunakan untuk menaktifkan pam tapisan, yang akan dijalankan sehingga bacaan penderia cahaya melebihi `DIRTY_THRESHOLD` untuk 60 saat. Anda dikehendaki menulis atur cara dalam **Bahasa Himpunan ARM** untuk sistem pembersihan akuarium tersebut.*

Anda tidak perlu mengambil kira konfigurasi sistem ataupun konfigurasi perkakasan terperinci yang lain. Kawalan pam tapisan dicapai melalui subrutin `start_filter_pump()` *dan* `stop_filter_pump()`. *Hanya kod Himpunan ARM untuk bahagian pengisytiharan data serta atur cara utama (main) untuk sistem pembersihan akuarium perlu ditunjukkan dalam jawapan anda.*

Nota: *Penderia Cahaya dan Motor Pam harus diawalkan dengan nombor port yang betul (ditakrif sebagai* `LIGHT_PORT` *dan* `PUMP_MOTOR_PORT`*) pada permulaan atur cara anda. Tambahan, lampu LED Penderia Cahaya harus diaktifkan secara menggunakan* `nx_sensors_analog_digi_set()`.

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 PUMP_MOTOR_PORT 2
#define LIGHT_PORT 0
#define LIGHT_LED_DATAPIN 0
#define DIRTY_THRESHOLD 87
#define SYSTICK_1000MS 1000
typedef unsigned long U32;
typedef unsigned char U8;

/** 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.
*/
void nx_sensors_analog_enable(U32 sensor);

/** Set the DIGI pin @a datapin of @a sensor.
*
* @param sensor The sensor port.
* @param datapin The DIGI pin to set.
*/
void nx_sensors_analog_digi_set(U32 sensor, U32 datapin);

/** Get a normalized analog reading in percentage from @a sensor.
*
* @param sensor The sensor port.
* @return An U8. The A/D value is converted to the range 0-100 %.
*
* @note The sensor port must be enabled in analog mode.
*/
U8 nx_sensors_analog_get_normalized(U32 sensor);
```

```
/** Initialize @a water filter pump.
*
* @param motor The pump motor port.
*/
void filter_pump_init(U32 motor);

/** Start the pump for the @a water filter.
*
* @param motor The pump motor port.
*/
void start_filter_pump(U32 motor);

/** Stop the pump for the @a water filter.
 *
 * @param motor The pump motor port.
*/
void stop_filter_pump(U32 motor);
```

(25 /100)