THE UNIVERSITY OF THE WEST INDIES

EXAMINATIONS OF DECEMBER 2010

Code and Name of Course: **ECNG3006 Microprocessor Systems** Paper:

Date and Time: **Thu 9th December 2010 9AM** Duration: **Three Hours**

INSTRUCTIONS TO CANDIDATES: This paper has 7 pages and 4 questions

Attempt ALL questions.
Questions 1-3 are each worth 20 marks.
Question 4 is worth 40 marks.
Maximum 100 marks.

This exam contributes 50% of the final course grade.
You must pass this exam in order to pass the course.

The following reference information is provided:

The graph paper on page 7 should be used to answer Q4.f.
Please attach to exam booklet at the end of the exam.

. . . continued

Q1. Kernels are used to facilitate **task management**.

(a) i. Draw a diagram to illustrate task states, and the functions that cause tasks to change state under a typical kernel. *4 marks*

ii. Identify two facilities present in real-time kernels, which are typically not present in general-purpose operating systems. Link the facilities identified to the requirements of real-Time systems. *4 marks*

(b) Explain how the uCOS-II task stack is manipulated during a context switch. Your response should differentiate between the processor stack, the C Run-time stack, and the task stack. *6 marks*

(c) Identify the number of tasks and the scheduling algorithm required for the system described on page 4. Based on these requirements, determine the number of unique priority levels needed, and justify a kernel choice between uCOS-II and freeRTOS. *6 marks*

Q2. The application platform used should possess certain **hardware features**.

(a) The μCOS-II kernel requires a platform processor that provides a user-accesible processor stack and an internal hardware timer that can generate interrupts. Explain why this is so. *4 marks*

(b) The magnetic card readers (as described on page 4) each require a 300 baud RS232 serial interface, but the PIC18 only has 1 hardware USART. Create a workable solution which allows the PIC18 to communicate with both magnetic card readers. Show all timing calculations/considerations for your proposed solution. *4 marks*

(c) Design appropriate circuitry which will allow the door locks (as described on page 4) to be activated from the PIC18. Show all current and voltage calculations/considerations, and estimate the average power consumption per lock, for your proposed solution. *12 marks*

Q3. A **Finite State Machine (FSM)** may be described using a state diagram. Two common FSM implementation techniques are switch-case statements, or a finite state table.

(a) Construct a state diagram for Task A (as described on page 4). *4 marks*

(b) Differentiate between the two strategies for implementing an FSM using the C language. Your answer should contrast strategies in terms of their resource requirements, and impacts on timing. *4 marks*

(c) MISRA-C Rule 15.2 states that "an unconditional break statement shall terminate every non-empty switch clause". Use a fragment of C-code to illustrate a violation of this rule. Predict how violation of this rule could impact upon a switch-case implementation of an FSM. *4 marks*

(d) Describe, and/or create, appropriate initialization, and FSM-based update routines for Task A (as described on page 4), using the C programming language, or C-like pseudo-code. You may presume appropriate μCOS-II kernel and C18 compiler library functions are available. *8 marks*

. . . continued

Q4. System behavior can be predicted using standard **performance metrics and tools**. For the system(as described on page 4):

(a) Identify the nominal task parameters (ϕ, p, e, D) for each Task A, B, F, R. Show your reasoning. *4 marks*

(b) Calculate the hyper-period of the system. Show your reasoning. *2 marks*

(c) Appropriately subdivide tasks into jobs, and identify the job parameters:

$$J_{Q,n} : e_{Q,n}(r_{Q,n}, d_{Q,n}]$$

for all job instances n belonging to individual tasks during the worst-case hyper-period. *8 marks*

(d) Draw the job precedence graph - use this graph to determine effective deadlines and release times. Show your reasoning. *10 marks*

(e) Calculate the system utilization. Show your reasoning. *2 marks*

(f) Using the Graph Paper on page 7, draw the task timelines from t=0 ms to t=1200 ms where: *8 marks*

- Card-data is available at both front and rear doors at t=0,
- Card read at the front door is at top of Task B search list,
- Card read at the rear door is invalid,
- Job priorities are determined using pre-emptive Earliest Deadline First scheduling,
- Job execution times are adjusted based on pre-emption overhead, and/or time spent waiting for/acquiring messages

Explain your reasoning.

(g) Based on the preceding analyses, critically assess system performance. Your answer should identify potential system hazards, and consider whether a pre-emptive priority based scheduler is suitable for the application. *6 marks*

END OF QUESTIONS

. . . continued

Case Information

A building entry system is designed around a PIC18 micro-controller, powered at 5Vdc, with a 200kHz clock. The system supports two entry/exit doors. There is a magnetic card reader and an electrically controlled lock at each door. A maximum of 1000 users may be issued with uniquely coded magnetic cards. An embedded kernel must be selected for the building entry system. The kernel must meet the following requirements:

- pre-emptive priority based scheduler with support for the Earliest Deadline First scheduling algorithm
- kernel tick interrupt every 10 milli-seconds (ms), with a nominal overhead of 1 ms
- allow the following tasks
  - Task A is responsible for serially polling each of the magnetic card readers every 0.4 seconds.
  - Task B is responsible for verifying whether a specific magnetic card is valid, and is allowed entry.
  - Tasks F and R are responsible for front/rear door-locks respectively - door locks are released for at least 5 seconds, but not more than 10 seconds before being re-activated.
- support inter-task resource usage, messaging and/or synchronization
  - Tasks F and R are synchronized with Task B; doors open only AFTER Task B has completed.
  - Task A sends 12 byte messages to Task B: (1 byte: door id, 10 bytes: card id, 1 byte: card id checksum)

Task Name	Operation and Estimated Times
A	job(s) released periodically request state of front door (1-5 ms) wait for response from front door (1-5 ms) if front door has a reading (1 ms) add front door id to buffer (1 ms) while front door has more chars (max 11 iterations) request char from front door (1-5 ms) wait for response from door (1-5 ms) add char to buffer (1 ms) else (1 ms) request state of rear door (1-5 ms) wait for response from door (1-5 ms) if rear door has a reading (1 ms) add rear door id to buffer (1 ms) while rear door has more chars (max 11 iterations) request char from rear door (1-5 ms) wait for response from door (1-5 ms) add char to buffer (1 ms) if buffer is not empty (1 ms) send 12 byte message to Task B (1 ms)
B	job(s) released on receipt of a message if reader identifier valid (1 ms) calculate checksum (10 ms) if checksum matches (1 ms) search list for checksum (1 - 1000 ms) if checksum found in list (1 ms) if reader is front (1 ms) send message to Task F (1 ms) if reader is rear (1 ms) send message to Task R (1 ms)
F, R	job(s) released periodically if message received from B (1 ms) open door lock (1 ms) else (1 ms) release door lock (1 ms)

. . . continued

μCOSII Message/Semaphore functions

`void *OSMboxAccept(OS_EVENT *pevent)`	Returns the contents of message if available. Does not suspend the calling job.
`OS_EVENT *OSMboxCreate(void *msg)`	Creates and initializes a mailbox
`void *OSMboxPend(OS_EVENT *pevent, INT16U timeout, INT8U *err)`	Suspends calling job until a message is available.
`INT8U OSMboxPost(OS_EVENT *pevent, void *msg)`	Sends a message to a job via the mailbox
`INT8U OSMboxQuery(OS_EVENT *pevent, OS_MBOX_DATA *pdata)`	Obtains information about a message mailbox
`INT16U OSSemAccept(OS_EVENT *pevent)`	Checks to see if a resource is available. Does not suspend the calling job if resource is not available.
`OS_EVENT *OSSemCreate(WORD value)`	Creates and initializes a semaphore
`void OSSemPend(OS_EVENT *pevent, INT16U timeout, INT8U *err)`	Suspends calling job until resource is available
`INT8U OSSemPost(OS_EVENT *pevent)`	Release indicated resource
`INT8U OSSemQuery(OS_EVENT *pevent, OS_SEM_DATA *pdata)`	Obtains information about a semaphore

C18 Library functions for RS232 Peripherals and Ports.

`void CloseUSART(void)`	Disable the USART.
`char DataRdyUSART(void)`	Is data available in the USART read buffer?
`char getcUSART(void)`	Read a byte from the USART.
`void OpenUSART(unsigned char config, unsigned int spbrg)`	Configure the USART.
`void putcUSART(char data)`	Write a byte to the USART.
`void baudUSART(unsigned char baudconfig)`	Set the baud rate configuration bits for enhanced USART.

Hardware information Manual Swipe Magnetic Card Reader

TTL-compatible interface (clock & data)

Recording method	FM decoding (F2F)
Card feeding speed	10 to 120 cm/second
Service life of magnetic head	300,000 passes
Operating power supply	+5 V dc ±10%
Current consumption	3mA typical; 5mA maximum
Mounting location	Indoors - no direct sunlight or humidity
Vibration Tolerance	10 to 150 Hz, single vibration width of 0.15 mm
Interface	RS232 - 2400 baud, 8 bits, no parity, 1 stop bit
Protocol	on receipt of the letter"Q" will respond with the number of characters in the last bar code scanned. On receipt of the letter "N" will respond with the next letter in the sequence of the last bar code scanned.

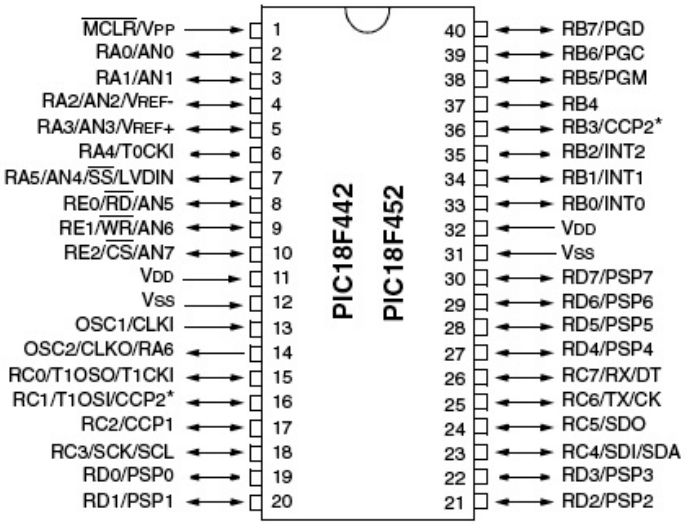
. . . continued

Hardware information – Electric bolt

Bolt collapses (opens) when power is applied.

Voltage	12Vdc
Pull-in Current	0.9 A
Holding Force	up to 1200 lbs (545 kg)

PIC18F452 description

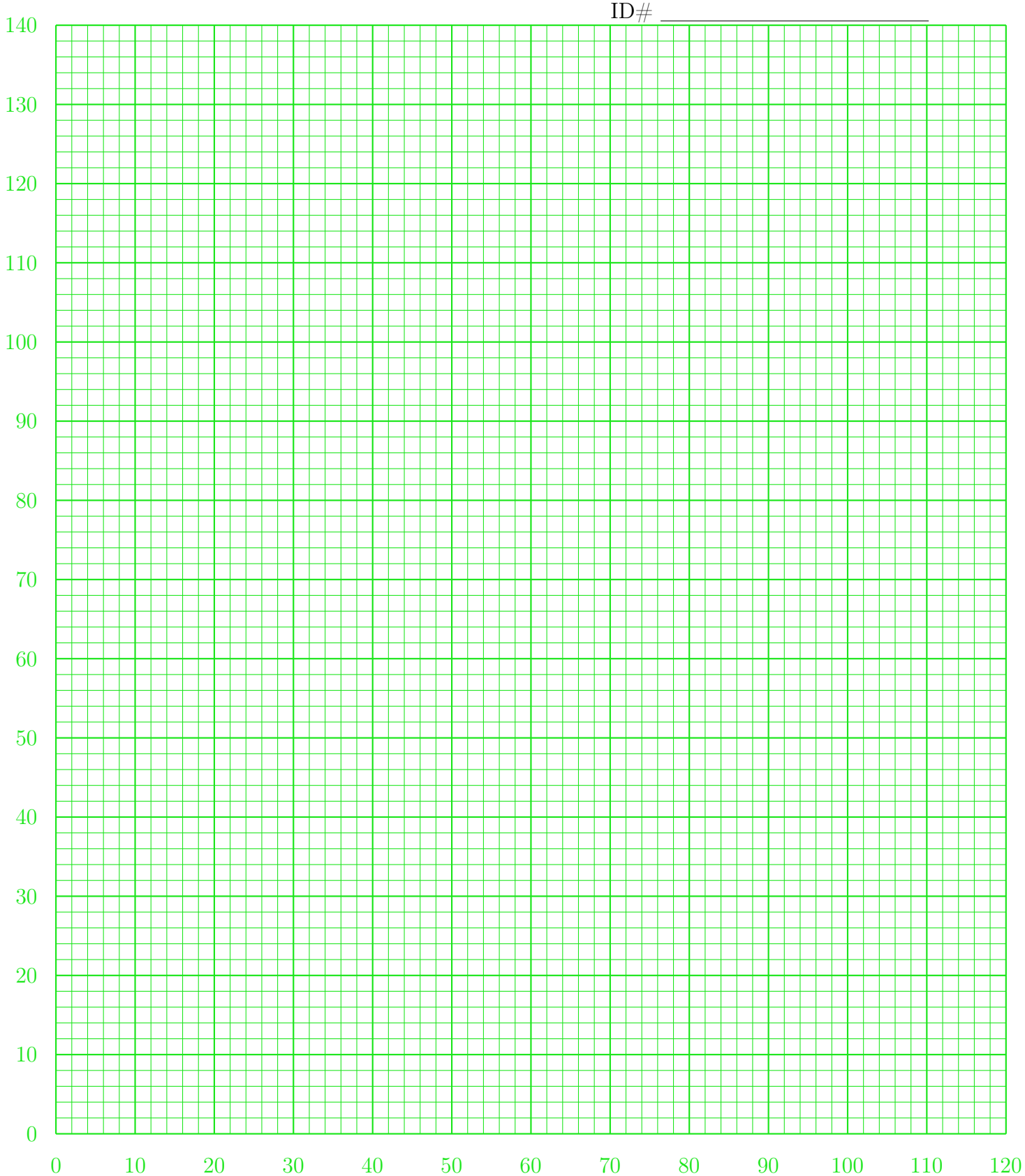


Output Pin Load resistance	R_L	464Ω
Output Pin Load capacitance	C_L	50 pF
Input Pin capacitance	C_{in}	5 pF
Max. Port output rise/fall time	T_{io}	40ns
Max. Delay port output	$T_{os}H2_{io}$	255ns
Min. Input Hold time	$T_{os}H2_{io}I$	100 ns
Max. current sunk/sourced by output (ignoring totals)		20mA
Max. input leakage current	I_{IL}	$\pm 1\mu$A
Output Low Voltage	V_{OL}	0.6V
Output High Voltage	V_{OH}	$V_{DD} - 0.7$ Volts
Max. Input Low Voltage (TTL)	V_{IL}	$0.15V_{DD}$ Volts
Min. Input High Voltage (TTL)	V_{IH}	$0.25V_{DD} + 0.8$ Volts
Max. Input Low Voltage (ST)	V_{IL}	$0.2V_{DD}$ Volts
Min. Input High Voltage (ST)	V_{IH}	$0.8V_{DD}$ Volts

All voltages must lie between V_{dd} and V_{ss}. All ratings taken at 25°. Pins on ports C,D and E are normally Schmitt Trigger inputs, push-pull outputs. Port A pin 4 is a Schmitt Trigger input, open-drain output. All other pins are normally TTL level inputs, push-pull outputs. Input circuitry on some pins changes depending on the pin function selected.

. . . continued

ID# _____



END OF QUESTION PAPER