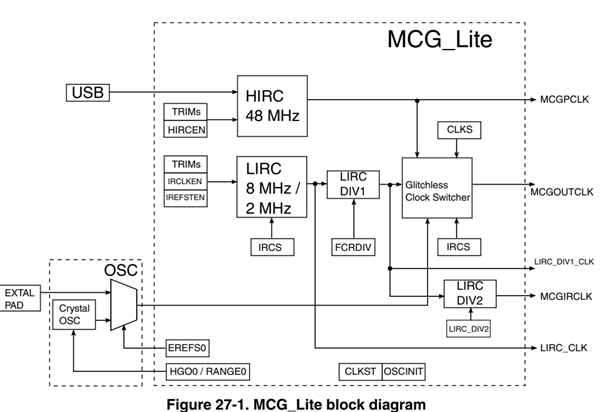
**CG2271 Realtime Operating Systems**

**Lab 3 – Timers**

**Answer Book**

|  |  |
| --- | --- |
| **Name** | **Student ID (Axxxxx)** |
| John Woo Yi Kai | A0272561E |
| Rishi Moorthy | A0273618X |
| Wang Yanjie | A0276202M |
| Kenneth Wong Cun Wi | A0303203A |
| Kuek Yeau Hao Jonathan | A0258485M |

**Question 1.** (2 MARKS)



As seen from the diagram above, we will not want to set the division factor of LIRC DIV1 to a value of 2 because this would change the output frequency of MCGOUTCLK, which undesirably affects others like the System Clock and Bus Clock. By setting LIRC DIV2 to 2 instead of LIRCDIV1, it would ensure that the System Clock and Bus Clock are unaffected while still obtaining the desired 1MHz signal from MCGIRCLK clock source.

**Question 2.** (4 MARKS)

// Configure the MCG Internal Reference Clock

void setMCGIRClk() {

// Choose LIRC as the clock source and enable LIRC

MCG->C1 &= ~MCG\_C1\_CLKS\_MASK;

MCG->C1 |= ((MCG\_C1\_CLKS(0b01) | MCG\_C1\_IRCLKEN\_MASK));

// Choose the 2 MHz clock

MCG->C2 &= ~MCG\_C2\_IRCS\_MASK;

// Set FRCDIV to dividing factor of 1

MCG->SC &= ~MCG\_SC\_FCRDIV\_MASK;

MCG->SC |= MCG\_SC\_FCRDIV(0b0);

// Choose LIRC\_DIV2 to dividing factor of 2

MCG->MC &= ~MCG\_MC\_LIRC\_DIV2\_MASK;

MCG->MC |= MCG\_MC\_LIRC\_DIV2(0b001);

}

**Question 3.** (2 MARKS)

Prescalar: 8

Modulo: 62500

**Question 4.** (5 MARKS)

void initTimer() {

// Disable TPM0 interrupt

NVIC\_DisableIRQ(TPM0\_IRQn);

// Initialize the MCG Internal Reference Clock

setMCGIRClk();

// Turn on the clock gating

SIM->SCGC6 |= SIM\_SCGC6\_TPM0\_MASK;

// Set clock source

SIM->SOPT2 &= ~SIM\_SOPT2\_TPMSRC\_MASK;

// Use MCGIRCLK

SIM->SOPT2 |= SIM\_SOPT2\_TPMSRC(0b11);

// Turn off TPM0 and clear the prescalar mask

TPM0->SC &= ~(TPM\_SC\_CMOD\_MASK | TPM\_SC\_PS\_MASK);

// Set the prescalar and the TOIE bit (prescalar = 8)

TPM0->SC |= ((TPM\_SC\_TOIE\_MASK) | TPM\_SC\_PS(0x3));

// Initialize the count to 0 (clearTPMx\_CNT b4 setting TPMx\_MOD)

TPM0->CNT = 0;

// Initialize modulo to create an

// interval of 500 ms.

TPM0->MOD=62500;

// Set priority to highest

NVIC\_SetPriority(TPM0\_IRQn,0);

// Enable TPM0 IRQ

NVIC\_EnableIRQ(TPM0\_IRQn);

}

**Question 5.** (2 MARKS)

void TPM0\_IRQHandler(){

// Clear pending IRQ

NVIC\_ClearPendingIRQ(TPM0\_IRQn);

if(TPM0->STATUS & TPM\_STATUS\_TOF\_MASK) {

count = (count + 1) % 6;

// Reset CNT to 0

TPM0->CNT = 0;

// Clear TOF bit

TPM0->STATUS |= TPM\_STATUS\_TOF\_MASK;

}

}

**Question 6.** (2 MARKS)

void startTimer() {

// Use TPM counter clock to increment

TPM0->SC |= TPM\_SC\_CMOD(0b1);

}

// Turn off the timer.

void stopTimer() {

TPM0->SC |= TPM\_SC\_CMOD(0b0); (this is wrong, does not clear the bit)

Should be:

TPM0->SC &= ~TPM\_SC\_CMOD(0b0);

}

**Report: \_\_\_\_\_\_\_\_\_\_\_\_\_/17**

**DEMO** (3 MARKS)

**Total: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / 20**