Inter-Task Communication (KeyPad & Motor)

조 진 성
경희대학교 컴퓨터공학과
Mobile & Embedded System Lab.

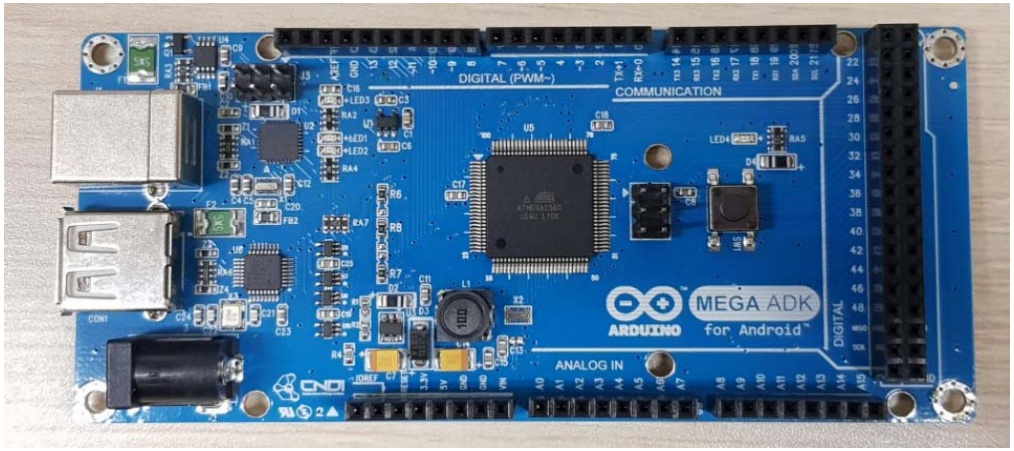
Inter-Task Communication

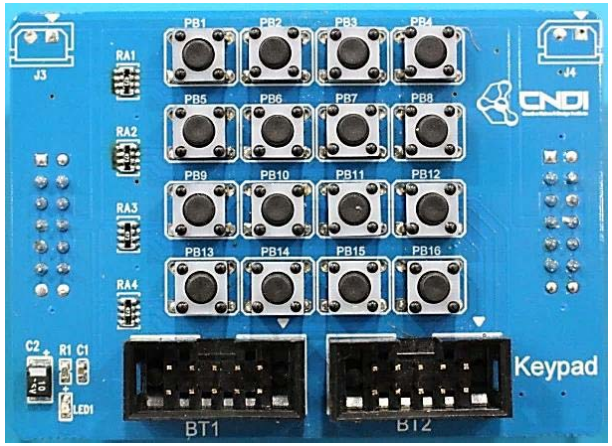
ITC in FreeRTOS

- Queue / Queue sets / Stream buffers / Message buffers
- Queue in FreeRTOS
  - xQueueHandle xQueueCreate(unsigned portBASE_TYPE uxQueueLength,
    unsigned portBASE_TYPE uxItemSize);
  - portBASE_TYPE xQueueSend(xQueueHandle xQueue,
    const void *pvItemToQueue,
    portTickType xTicksToWait);
  - portBASE_TYPE xQueueSendFromISR(xQueueHandle xQueue,
    const void *pvItemToQueue,
    BaseType_t *pxHigherPriorityTaskWoken);
  - portBASE_TYPE xQueueReceive(xQueueHandle xQueue,
    void *pvBuffer,
    portTickType xTicksToWait);
  - portBASE_TYPE xQueueReceiveFromISR(xQueueHandle xQueue,
    void *pvBuffer,
    BaseType_t *pxHigherPriorityTaskWoken);

하드웨어 구성

❖ **KeyPad를 통해 Motor 제어**





하드웨어 구성

❖ KeyPad를 통해 Motor 제어

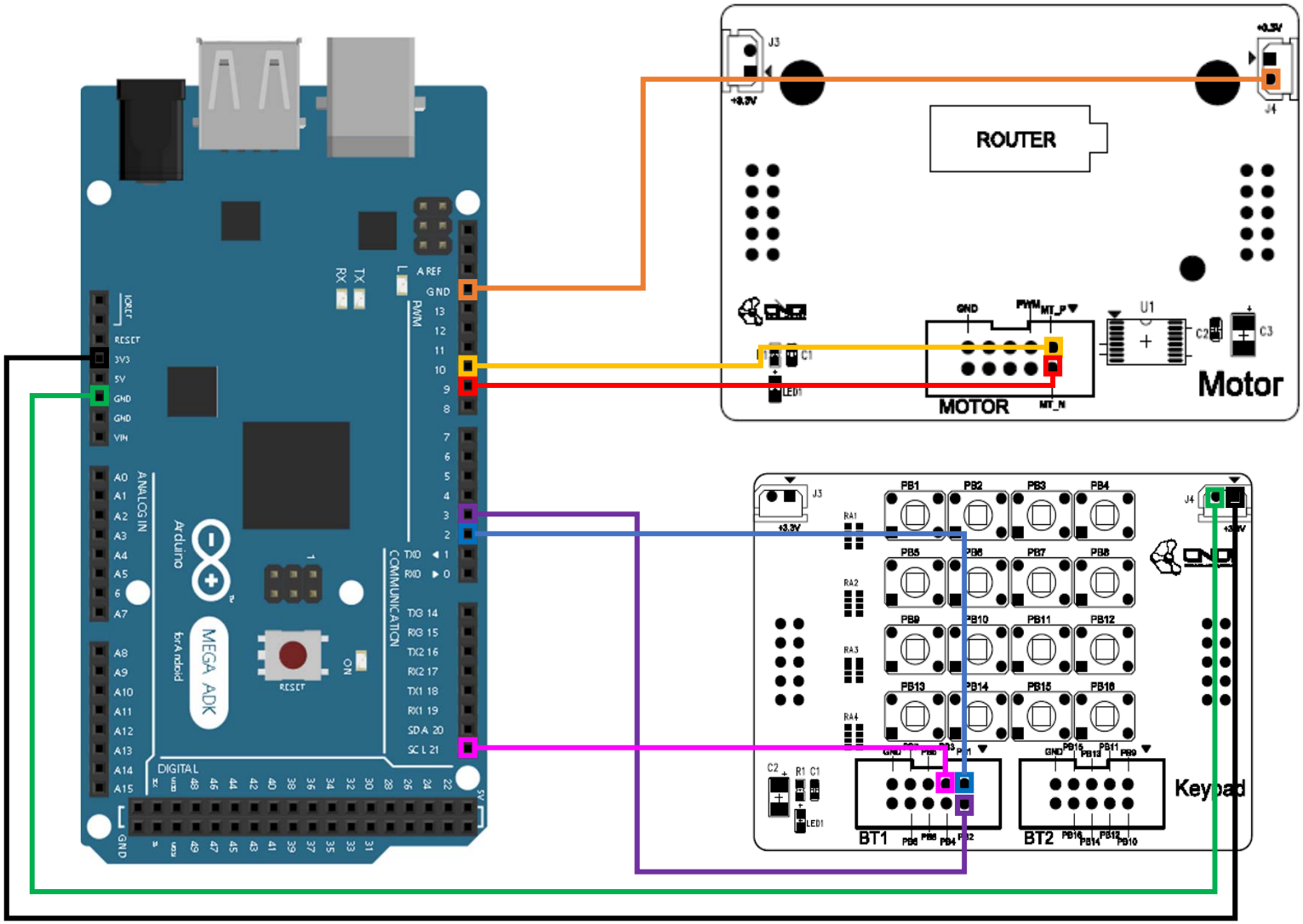
- 왼쪽 회전, 정지, 오른쪽 회전 3개 키에 의해 모터 동작
- KeyPad ISR과 Motor Task간 Queue를 이용해 데이터 전달

- KeyPad를 통해 Motor 제어하기 위한 하드웨어 구성
  - Arduino MEGA ADK의 DIGITAL 2*(Interrupt 0)*와 KeyPad 모듈의 BT1 포트의 핀 PB1을 연결
  - Arduino MEGA ADK의 DIGITAL 3*(Interrupt 1)*와 KeyPad 모듈의 BT1 포트의 핀 PB2을 연결
  - Arduino MEGA ADK의 DIGITAL 21*(Interrupt 2)*와 KeyPad 모듈의 BT1 포트의 핀 PB3을 연결
  - Arduino MEGA ADK의 3.3V와 KeyPad 모듈의 J4 포트의 +3.3V와 연결
  - Arduino MEGA ADK의 GND(0V)와 KeyPad 모듈의 J4 포트의 좌측에 연결

  - Arduino MEGA ADK의 DIGITAL 10*(PWM 가능 핀)*와 Motor 모듈의 핀 MT_P를 연결
  - Arduino MEGA ADK의 DIGITAL 9*(PWM 가능 핀)*과 Motor 모듈의 핀 MT_N을 연결
  - Arduino MEGA ADK의 GND(0V)와 Motor 모듈의 J4 포트의 아래에 연결

하드웨어 구성

❖ **KeyPad를 통해 Motor 제어**



실습 예제

❖ KeyPad를 통해 Motor 제어

- lab3-3.ino

```
#include <FreeRTOS_AVR.h>

const int MT_P = 10;
const int MT_N = 9;

const int LeftKey   = 2;
const int StopKey  = 3;
const int RightKey = 21;

QueueHandle_t xQueue;

// ISR, Queue로 데이터를 전달
void LeftKeyControl() {
  uint16_t sendValue = 1;
  xQueueSendFromISR(xQueue, &sendValue, 0);
}

void StopKeyControl() {
 uint16_t sendValue = 2;
  xQueueSendFromISR(xQueue, &sendValue, 0);
}

void RightKeyControl() {
 uint16_t sendValue = 3;
  xQueueSendFromISR(xQueue, &sendValue, 0);
}
```

```
void MotorTask(void * arg) {
  uint16_t receiveValue = 0;

  while(1) {
    // Queue로 부터 데이터를 받음
    if( xQueueReceive(xQueue, &receiveValue, 0) ) {
        // Rotate left
        if(receiveValue == 1) {
          digitalWrite(MT_P, LOW);
          digitalWrite(MT_N, HIGH);
        }
        // Stop
        else if(receiveValue == 2) {
          digitalWrite(MT_P, LOW);
          digitalWrite(MT_N, LOW);
        }
        // Rotate right
        else if(receiveValue == 3) {
          digitalWrite(MT_P, HIGH);
          digitalWrite(MT_N, LOW);
        }
    }
  }
}
```

실습 예제

❖ KeyPad를 통해 Motor 제어

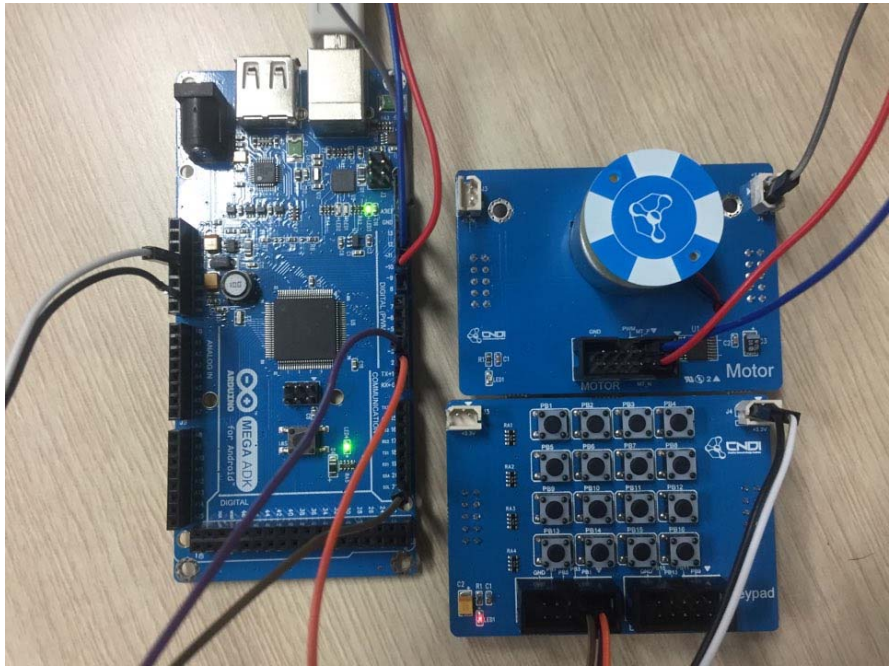
- lab3-3.ino

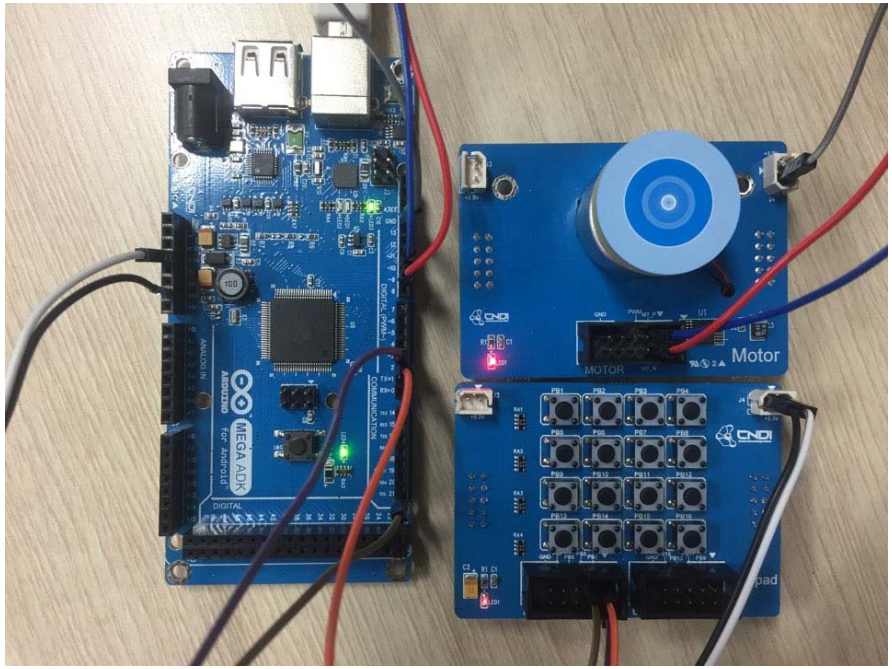
```
void setup() {
  pinMode(MT_P, OUTPUT);
  pinMode(MT_N, OUTPUT);

  pinMode(LeftKey, INPUT);
  pinMode(StopKey, INPUT);
  pinMode(RightKey, INPUT);

  // 인터럽트 설정, lab1-3 참고
  attachInterrupt(0, LeftKeyControl, RISING);
  attachInterrupt(1, StopKeyControl, RISING);
  attachInterrupt(2, RightKeyControl, RISING);

  xQueue = xQueueCreate(3, sizeof(uint16_t));

  if(xQueue != NULL) {
    xTaskCreate(MotorTask, NULL, 200, NULL, 1, NULL);
    vTaskStartScheduler();
  }
}

void loop() {
}
```

실습 예제

❖ KeyPad를 통해 Motor 제어

- 동작 화면





실습 과제

❖ 실습 예제 추가 구현

- 속도 제어 버튼 추가하기
  - 왼쪽 회전 또는 오른쪽 회전은 25% 회전
  - 버튼 2개를 추가하여 속도를 제어 (각각 왼쪽 또는 오른쪽)
  - 한번 누를 때마다 50% → 75% → 100% → 50%로 속도 제어

- 참고
  - Arduino에서 Motor 속도 제어하기
    - Arduino는 GPIO 핀으로 Analog Input은 가능하지만, 출력 전압을 조절하는 Analog Output은 불가능
      - Raspberry Pi는 Analog Input/Output 모두 불가능
    - 따라서, PWM(Pulse Width Modulation)을 통해 제어
    - analogWrite()를 통한 PWM 파형 생성
      - void analogWrite(uint8_t pin, int dutyCycle)
      - pin: PWM이 가능한 Digital Pin*(e.g. ~9, ~10, etc)* 사용
      - dutyCycle: 0(최저)~255(최고)

Q & A

http://mesl.khu.ac.kr