Mengirimkan Struct dengan Queue

Umumnya data tidak dikirimkan secara langsung pada queue, namun terlebih dahulu di masukkan kedalam sebuah struct. Ini berguna ketika data yang dikirimkan berbentuk objek yang memiliki beberapa atribut, contohnya ketika mengirimkan data suhu, kelembapan, beserta waktu saat ini dalam satu buah struct.

Pass Struct by Value

Jika struct dikirimkan by value ke dalam queue, maka FreeRTOS akan menyalin seluruh isi struct ke dalam buffer queue. Cara ini sederhana dan aman, tetapi untuk struct berukuran besar, proses copy bisa memperlambat sistem dan memperboros RAM untuk melakukan proses duplikasi.

```
//Send
Data_t data;
data.tick = xTaskGetTickCount();
snprintf(data.msg, sizeof(data.msg), "Hello Value");
xQueueSend(queue, &data, portMAX_DELAY);

// Receive
Data_t received;
xQueueReceive(queue, &received, portMAX_DELAY);
Serial.println(received.msg);
```

Pass Struct By Reference

Dalam hal ini, pointer memiliki peran penting, dimana dengan pass by reference, kita hanya mengirimkan alamat memori dari struct tersebut ke dalam queue, bukan seluruh isi datanya sehingga data tidak perlu disalin ulang ke dalam buffer queue, sehingga lebih efisien dalam penggunaan memori.

```
// Send
Data_t *data = (Data_t *) pvPortMalloc(sizeof(Data_t));
data->tick = xTaskGetTickCount();
snprintf(data->msg, sizeof(data->msg), "Hello Pointer");
```

```
xQueueSend(queue, &data, portMAX_DELAY);

// Receive
Data_t *received;
xQueueReceive(queue, &received, portMAX_DELAY);
Serial.println(received->msg);
vPortFree(received); // Don't Forget
```

Pass by reference lebih direkomendasikan karena efisiensi dan performanya, namun dalam proses melakukan pass by reference penggunaan pointer perlu diteliti sehingga pengiriman value / memory tidak salah dan memory yang telah dialokasi di free.

Contoh Penggunaan

Berikut merupakan contoh aplikasi Struct dan Queue untuk mengirimkan tick dan pesan dari serial monitor dari suatu task, untuk ditampilkan pada task lainnya dengan format tertentu.

```
/* In this FreeRTOS example, we use xTaskCreatePinnedToCore()
   to show queue communication between tasks.
   We'll send both a serial message and a tick count via a struct pointer. */

#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "Arduino.h"

// Handle untuk Queue
QueueHandle_t xQueue;

// Ukuran buffer pesan
#define MSG_MAX_LEN 50

// Struct untuk data pesan
struct Message {
  TickType_t ticks;
  char text[MSG_MAX_LEN];
};

// Task untuk membaca input serial sampai newline
void SerialReadTask(void *pvParameters) {
  while (1) {
    if (Serial.available() > 0) {
```

```
      // Baca string sampai newline
      String input = Serial.readStringUntil('\n');

      if (input.length() > 0 && input.length() < MSG_MAX_LEN) {
        // Alokasikan memori untuk struct
        Message *msg = (Message *)pvPortMalloc(sizeof(Message));
        if (msg != NULL) {
          msg->ticks = xTaskGetTickCount();
          input.toCharArray(msg->text, MSG_MAX_LEN);

          // Kirim pointer ke queue
          if (xQueueSend(xQueue, &msg, portMAX_DELAY) == pdPASS) {
            Serial.println("[Sent to Queue]");
          } else {
            // Jika gagal, bebaskan memori
            vPortFree(msg);
          }
        }
      }
    }
    vTaskDelay(10 / portTICK_PERIOD_MS);
  }
}

// Task untuk membaca dari queue dan menampilkan pesan
void SerialPrintTask(void *pvParameters) {
  Message *received;

  while (1) {
    if (xQueueReceive(xQueue, &received, portMAX_DELAY) == pdPASS) {
      Serial.print("Message received after ");
      Serial.print((unsigned long)received->ticks);
      Serial.print(" ticks: \"");
      Serial.print(received->text);
      Serial.println("\"");

      // Bebaskan memori setelah dipakai
      vPortFree(received);
    }
  }
}
```

```
void setup() {
  Serial.begin(115200);
  delay(1000);

  // Buat queue untuk menampung 5 pointer ke Message
  xQueue = xQueueCreate(5, sizeof(Message *));
  if (xQueue != NULL) {
    xTaskCreatePinnedToCore(SerialReadTask, "SerialRead", 4096, NULL, 1, NULL, 0);
    xTaskCreatePinnedToCore(SerialPrintTask, "SerialPrint", 4096, NULL, 1, NULL, 1);
  } else {
    Serial.println("Error: Queue creation failed!");
  }
}

void loop() {
}
```

Revision #1
Created 2025-09-13 19:05:06 UTC by Digilab UI
Updated 2025-09-13 19:05:38 UTC by Digilab UI