

Họ và tên: Thiều Quang Anh Tuấn

MSSV: 21207239

Báo cáo thực hành

1. Xây dựng ứng dụng đếm số lượng ký tự xuất hiện nhiều nhất: Người dùng từ PC gửi một đoạn văn đến EFR32xG21 thông qua UART, EFR32xG21 kiểm tra và trả về ký tự xuất hiện nhiều nhất theo định dạng “ký tự: số lượng ký tự đó”, ví dụ ký tự “a” xuất hiện nhiều nhất là 22 lần, kết quả trả về cho PC là “a: 22”.

Yêu cầu:

- Sử dụng DMA để di chuyển dữ liệu từ Rx Buffer của USART đến Memory.
- Vì số lượng dữ liệu phản hồi về PC không quá lớn nên dùng CPU để cập nhật dữ liệu cho Tx Buffer.

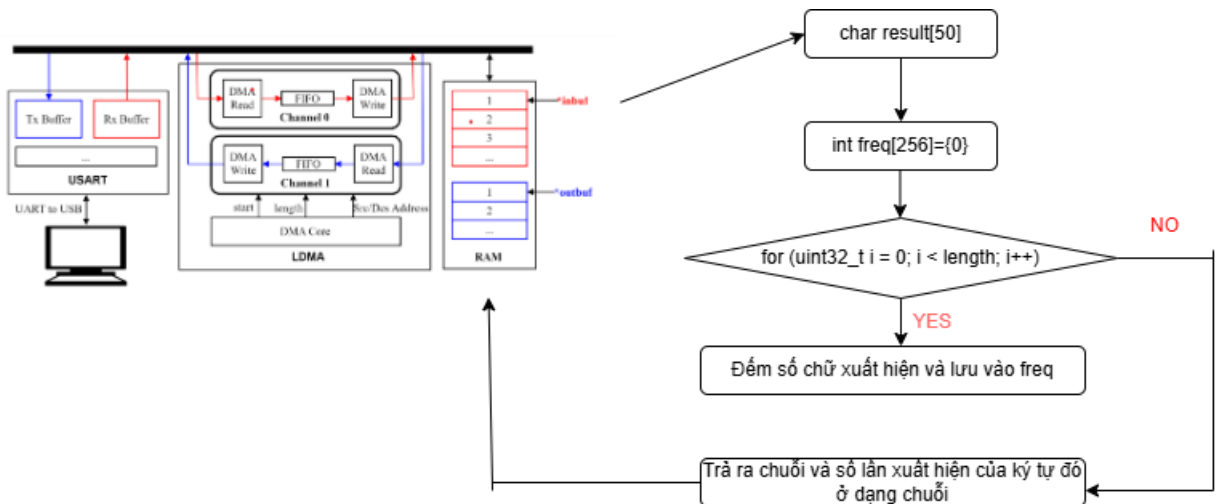
- **Ý tưởng làm bài:**

- Nhận dữ liệu từ PC truyền xuống UART, sau khi nhận xong chúng ta bắt đầu sử dụng dữ liệu đã nhận được để xử lý.

- CPU đọc dữ liệu từ vùng nhớ mà DMA đã lưu, duyệt qua từng ký tự và sử dụng mảng array để đếm số lần ký tự đó xuất hiện, tiếp theo là xác định ký tự xuất hiện nhiều nhất.

- Sau khi có ký tự xuất hiện nhiều nhất ta sẽ ép kiểu sang dạng chuỗi để UART tự động truyền từng byte qua giao tiếp UART đến PC.

- **Flow chart:**



- **Đoạn code cho chương trình trên:**

Tham khảo tại Github:

https://github.com/TQAT0707GW/BaoCaoThucHanhGTMT/blob/main/Tuan_4/Lab4_Cau1.c

Giải thích sơ lược về đoạn code chính:

Đối với các hàm

```
void countCharacters(uint8_t *buffer, uint32_t length, char *result) {  
    int freq[256] = {0}; // Lưu số lần xuất hiện của mỗi ký tự ASCII  
  
    // Đếm số lần xuất hiện của từng ký tự  
    for (uint32_t i = 0; i < length; i++) {  
        freq[buffer[i]]++;  
    }  
  
    // Tìm ký tự xuất hiện nhiều nhất  
    char maxChar=0; int i=1;  
    int maxCount = freq[0];  
    for (i; i < 256; i++) {  
        if (freq[i] > maxCount) {  
            maxCount = freq[i];  
            maxChar = (char)i;  
        }  
    }  
  
    // Trả kết quả dạng "ký tự: số lần"  
    sprintf(result, " %c: %d ", maxChar, maxCount);  
    // Chuyển từ số sang chuỗi  
-}
```

Đếm số lần xuất hiện đoạn của các ký tự. Ví dụ ký tự a có mã ASCII=48 thì ta tăng cái mảng với thứ tự 48 lên 1 đơn vị.

Đoạn code này để tìm kiếm ký tự xuất hiện nhiều nhất. Đồng thời khi tìm kiếm xong ta sẽ ép kiểu nó thành char và lưu vào maxChar

Đoạn code này dùng để ép kiểu từ số sang chuỗi dùng để gửi lên USART, sau đó trả kết quả về con trỏ result (bao gồm ký tự xuất hiện nhiều lần nhất và số lần xuất hiện)

Đối với hàm main:

```
while (1) {  
    sl_system_process_action();  
    app_process_action();  
  
    rx_done = false;  
  
    LDMA_StartTransfer(RX_LDMA_CHANNEL, &ldmaRXConfig, &ldmaRXDescriptor);  
  
    while (!rx_done) {  
        EMU_EnterEM1();  
    }  
  
    // Xử lý dữ liệu nhận được  
    char result[50]; // lưu kết quả sau khi xử lý chuỗi  
    countCharacters(inbuf, BUFLen, result);  
  
    for (uint32_t i = 0; i < strlen(result); i++) {  
        USART_Tx(USART0, result[i]);  
        while (!(USART_StatusGet(USART0) & USART_STATUS_TXBL));  
    }  
}
```

Bắt đầu quá trình DMA từ Rx buffer đến inbuf

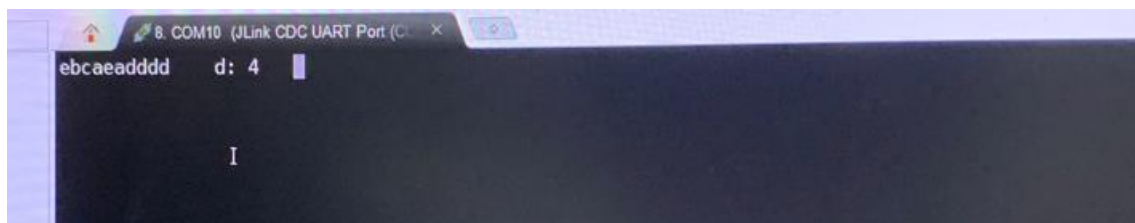
Chờ DMA kết thúc

Chuyển MCU vào mode EM1 để sử dụng interrupt

Gửi dữ liệu vào hàm để lấy ra kết quả chuỗi

Sau đó ta dùng CPU để truy cập và gửi lên USART

Kết quả:



Sau khi xong thì ta thấy kết quả giống như yêu cầu mà đề đặt ra. Trong bài này, khi gửi gửi 10 ký tự lần lượt là “ebcaeadddd” thì kết quả nhận được như hình với ký tự ‘d’ xuất hiện nhiều nhất là 4 lần.

2. Xây dựng ứng dụng đo thời gian DMA từ Memory đến UART, với ba số lượng dữ liệu khác nhau: 10 Byte, 100 Byte và 1000 Byte. Dữ liệu truyền sẽ được tạo ra tùy ý (có thể là dữ liệu ngẫu nhiên hoặc theo định dạng cụ thể). Người dùng gửi lệnh đo đến EFR32xG21 qua UART, sau đó EFR32xG21 sẽ phản hồi lại với dữ liệu đã tạo và thời gian DMA.

Yêu cầu:

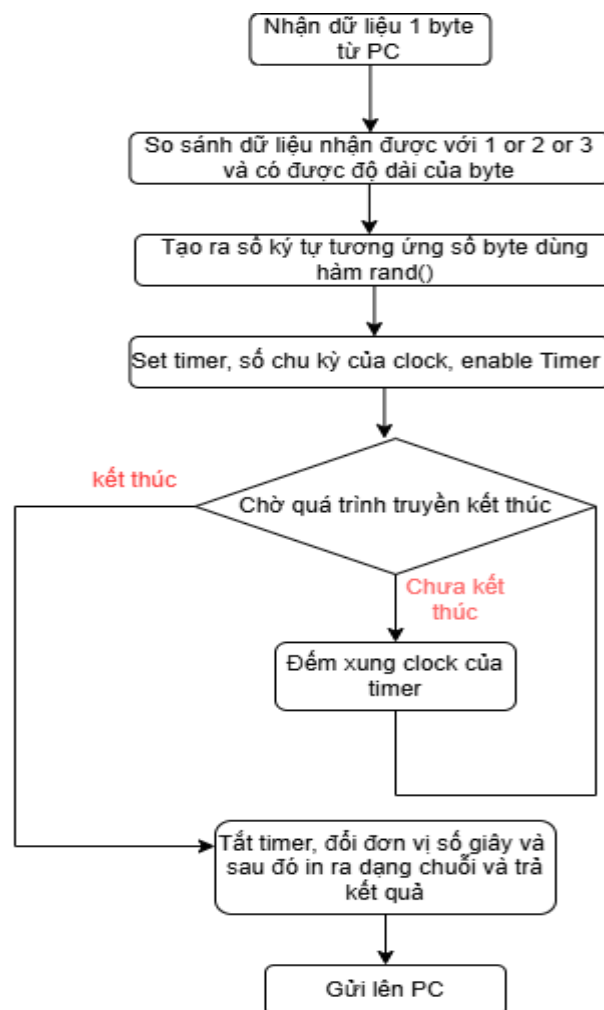
- Sử dụng DMA để di chuyển dữ liệu từ Memory đến UART.
- Sử dụng CPU để lấy yêu cầu từ PC.

- **Ý tưởng làm bài:**

-Nhận dữ liệu từ PC sau đó so sánh với từng ký tự, nếu 1 là 10 byte, nếu 2 là 100 byte và nếu 3 là 1000 byte. Từ đó tạo ra số ký tự với số byte tương ứng.

-Sau đó, đưa vào bộ đếm counter của Timer tính thời gian để truyền từ DMA lên PC và trả kết quả lên PC

- **Flow chart:**



- Đoạn code chương trình:

Github:

https://github.com/TQAT0707GW/BaoCaoThucHanhGTMT/blob/main/Tuan_4/Lab4_Cau2.c

Một số ý chính trong đoạn code:

Dòng 78-93

```

76
77 uint32_t calculatePeriod(uint32_t numClk)
78 {
79     uint32_t timerClockMHz = CMU_ClockFreqGet(CMU_Clock_TIMER0) / 1000000;
80     // Convert the count between edges to a period in microseconds
81     return (numClk / timerClockMHz);
82 }
83 uint32_t measureDMA(uint8_t *data, uint32_t length) {
84     uint32_t numClk = 0;
85     uint32_t elapsedTime = 0;
86     TIMER_CounterSet(TIMER0, 0);
87     TIMER_Enable(TIMER0, true);
88     startDMA(data, length);
89     while (!dma done);
90     TIMER_Enable(TIMER0, false);
91     numClk = TIMER_CounterGet(TIMER0);
92     elapsedTime = calculatePeriod(numClk);
93     return elapsedTime;

```

Hàm này trả về kết quả đơn vị của giây là (us)

Hàm measureDMA với 3 dòng code đầu khởi tạo timer và các thông số để đếm

Quá trình truyền DMA bắt đầu

Đợi quá trình truyền dma hoàn thành

Lấy dữ liệu từ timer và tính toán

Dòng 61-66

```

61 void startDMA(uint8_t *data, uint32_t length) {
62     dma_done = false;
63     ldmaTXDescriptor = (LDMA_Descriptor_t)LDMA_DESCRIPTOR_SINGLE_M2P_BYTE(data, &(USART0->TXDATA), length);
64     ldmaTXConfig = (LDMA_TransferCfg_t)LDMA_TRANSFER_CFG_PERIPHERAL(LdmaPeripheralSignal_USART0_TXBL);
65     LDMA_StartTransfer(TX_LDMA_CHANNEL, &ldmaTXConfig, &ldmaTXDescriptor);
66 }

```

Hàm này dùng để truyền DMA lên PC khi nào truyền xong thì dma_done=true

```

int main(void) {
    sl_system_init();
    initGPIO();
    initUSART0();
    initLDMA();
    initTimer();

    while (1) {
        if (USART_StatusGet(USART0) & USART_STATUS_RXDATAV) {
            inbuf[0] = USART_Rx(USART0); //nhận dữ liệu từ UART

            uint32_t length = 0;
            if (inbuf[0] == '1') {
                length = BUFFER_SIZE_10;
            } else if (inbuf[0] == '2') {
                length = BUFFER_SIZE_100;
            } else if (inbuf[0] == '3') {
                length = BUFFER_SIZE_1000;
            } else {
                continue;
            }

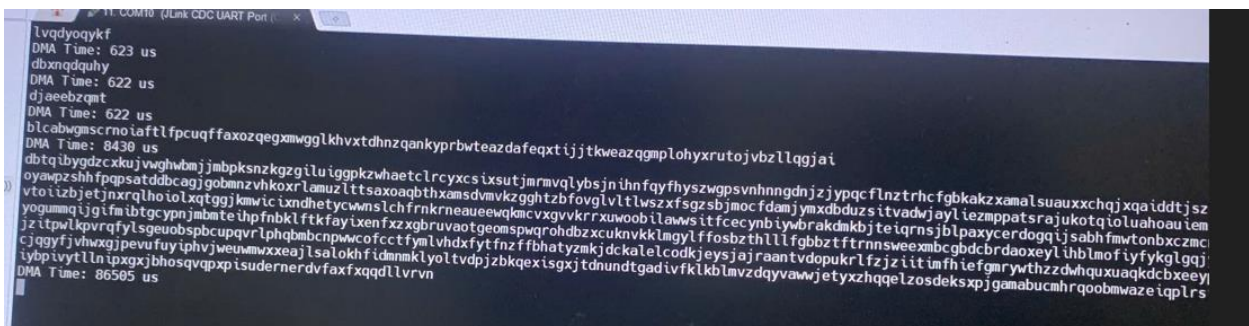
            generateRandomData(outbuf, length);
            uint32_t dmaTime = measureDMA(outbuf, length);

            char result[50];
            sprintf(result, "\r\nDMA Time: %lu us\r\n", dmaTime);
            startDMA((uint8_t *)result, strlen(result));
        }
    }
}

```

Trong hàm main này ta chủ yếu tập trung vào phần yêu cầu đề bài. Sau khi có kết quả chúng ta sẽ chuyển đổi sang ký tự và gửi lên PC.

• Kết quả:



Kết quả là thời gian truyền đo được trên mobaXterm, ta thấy khi gửi 10 byte thì thời gian là :622us, 100 byte thì 8430 us, còn 1000 byte thì 86585us

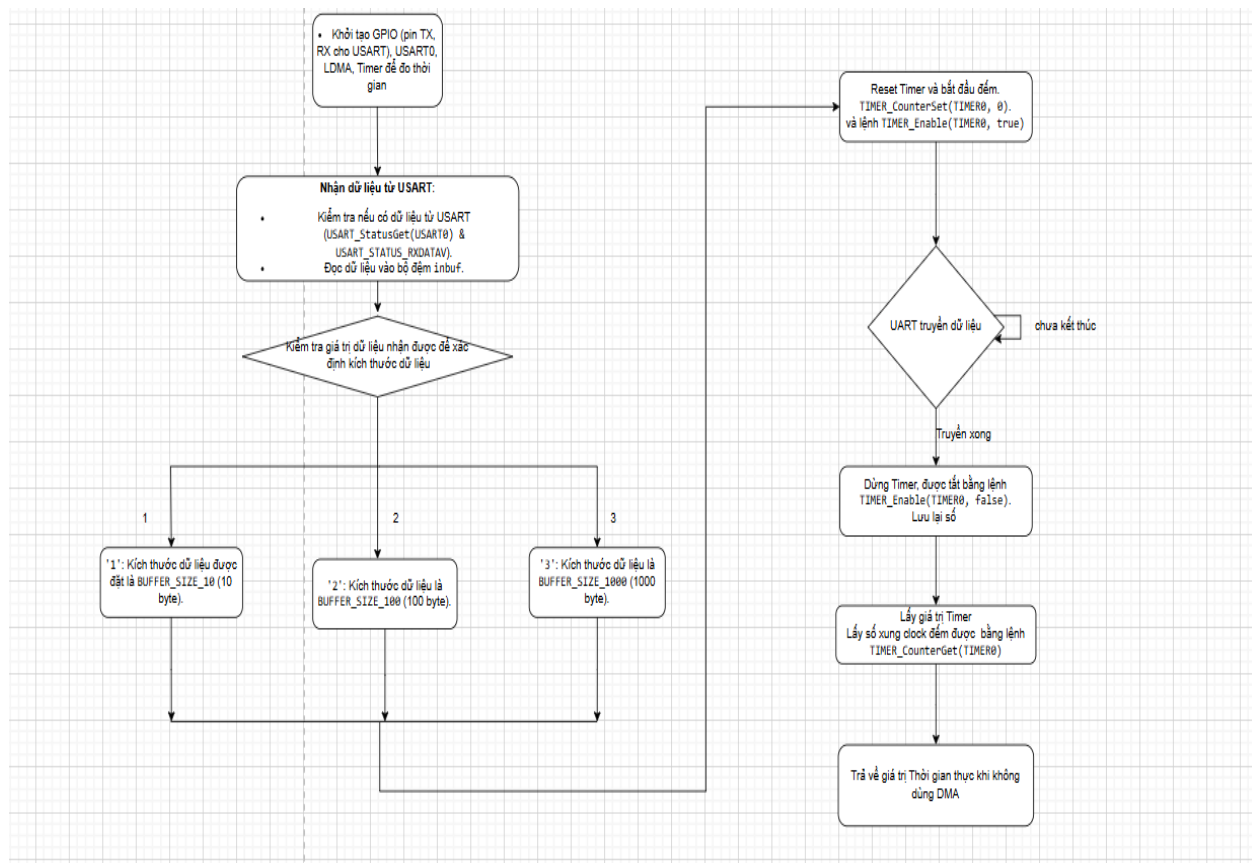
Câu 3: Xây dựng chương trình con tương tự như câu 2 nhưng không sử dụng DMA. So sánh và nhận xét kết quả so với câu 2.

- **Ý tưởng làm bài:**

-Nhận dữ liệu từ PC sau đó so sánh với từng ký tự, nếu 1 là 10 byte, nếu 2 là 100 byte và nếu 3 là 1000 byte. Từ đó tạo ra số ký tự với số byte tương ứng.

-Sau đó, đưa vào bộ đếm counter của Timer tính thời gian để truyền từ UART lên PC và trả kết quả lên PC

- **Flow chart:**



- **Đoạn code cho chương trình trên:**

- **Github:**

https://github.com/TQAT0707GW/BaoCaoThucHanhGTMT/blob/main/Tuan_4/Lab4_Cau3.c

- Một số ý chính của đoạn code:

```

3
4 uint32_t measureNoDMA(uint8_t *data, uint32_t length) {
5     TIMER_CounterSet(TIMER0, 0);
6     TIMER_Enable(TIMER0, true);
7
8     for (uint32_t i = 0; i < length; i++) {
9         USART_Tx(USART0, data[i]);
10        while (!(USART_StatusGet(USART0) & USART_STATUS_TXBL));
11    }
12
13    TIMER_Enable(TIMER0, false);
14    return TIMER_CounterGet(TIMER0);
15 }

```

Đoạn code này dùng để bật timer và tắt timer khi kết thúc

Đoạn code dùng để gửi lên bằng USART, chờ cho đến khi truyền xong

Hàm main:

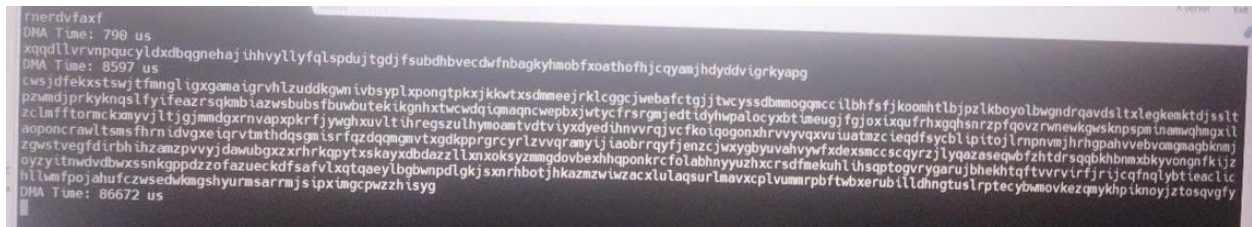
```

1 int main(void) {
2     sl_system_init();
3     initGPIO();
4     initUSART0();
5     initLDMA();
6     initTimer();
7
8     while (1) {
9         if (USART_StatusGet(USART0) & USART_STATUS_RXDATAV) {
10             inbuf[0] = USART_Rx(USART0);
11
12             uint32_t length = 0;
13             if (inbuf[0] == '1') {
14                 length = BUFFER_SIZE_10;
15             } else if (inbuf[0] == '2') {
16                 length = BUFFER_SIZE_100;
17             } else if (inbuf[0] == '3') {
18                 length = BUFFER_SIZE_1000;
19             } else {
20                 continue;
21             }
22
23             generateRandomData(outbuf, length);
24             uint32_t dmaTime = measureNoDMA(outbuf, length);
25             char result[50];
26             sprintf(result, "\r\nDMA Time: %lu us\r\n", dmaTime);
27             startDMA((uint8_t *)result, strlen(result));
28         }
29     }
30 }

```

Hàm main giống hoàn toàn với câu 2. Khác là chúng ta không đổi đơn vị và lấy đơn vị là us luôn.

- **Kết quả:**



```
rnerdvfaxf
DMA Time: 790 us
xqqdllrvnpqacyldxdbggnehaj uhhvlyllyqlspduj tgdj fsubdhbvecdwfnbagkyhmobfxoathofhj cqyamjhdyydivgrkyapg
DMA Time: 8597 us
cwsjdfekxstswjtfmngligxgamaigrvhlzuddkgm lvbsyplxpongtpkxjkkwtxsdmeejrkclggcjwebafctgjjtwcyssdbmmoggmccilbhfsfjkoamhtlbpzlkboylbwgnrqavds ltxlegkemtjdjssilt
pzwmddjprkyknqslfyifeazrsqkmbiazwsbubsfbunbutekikgnhxtwcwdqimaqncwepbxjwtycfrsrgmjedt idyhwpalocyxxt imeugj fgojoxlqufrhxgghsnrzpfqovzrnewkgwsknpspm tnamqhgxiil
zclmfftorackxmyvjltjgjmmdgxrnvapxpkrfjywhxuvlt thregszulhymoamtvdvixdyedihnvvrqjvc fkoigogonxhrrvvyvqxvu lueatazcleqdfsybl ipittojlnpnmvjhrhgpahvvebvomgmagbkmmj
aoponcrawltsmsfhrnidvgxeigrvtmthdsgm isrfqzqqgmmtxgdkpprgrcyr lzvvqamytj taobr rgyfjenzcyxygyuvahvyvfxdxsmccscqyrzj lyqazasequbfzhtdrsqgbkbbmxbkyvongn fkljz
zgwstvegfdirbh izamzpvvyj dawubgxzxrhkapytkskayxbddazllxn xoksyzmqgdovbexhqponkrcfolabhnyyuzhxcrcsdfmekuhl thsogtogurygarujbhekhqtftvrvirfjrijcqnqlybtieaclic
pyzyitnwdvdbwxssnkgppdz zofazueckdfsafvlxqtqae ylbgnw pdlgkjsxn rhbotjhkazmzv i wzacxluqsurlaavxcplvummrpbfwbxerubillahngtuslrptecybmmovkezqaykhpiknoy jztosqvgfy
hllwmpojahufczwsedwkmgshyurmsar majs ipxlmcpwz zhsy
DMA Time: 86672 us
```

- **Kết quả cho thấy rằng:**

- + Khi truyền 10 byte đơn vị là 790us
- + Khi truyền 100 byte là 8597us
- + Khi truyền 1000 bytes thì 86672us

Như ta thấy, thì câu 2 tốc độ gửi sẽ nhanh hơn câu 3 nhiều. Vì DMA thường sẽ nhanh hơn với các dữ liệu lớn, ngoài ra DMA là một khối hoạt động độc lập với CPU. Còn đối với khi CPU dùng USART gửi lên thì nó sẽ tốn thời gian do vừa đọc và ghi.