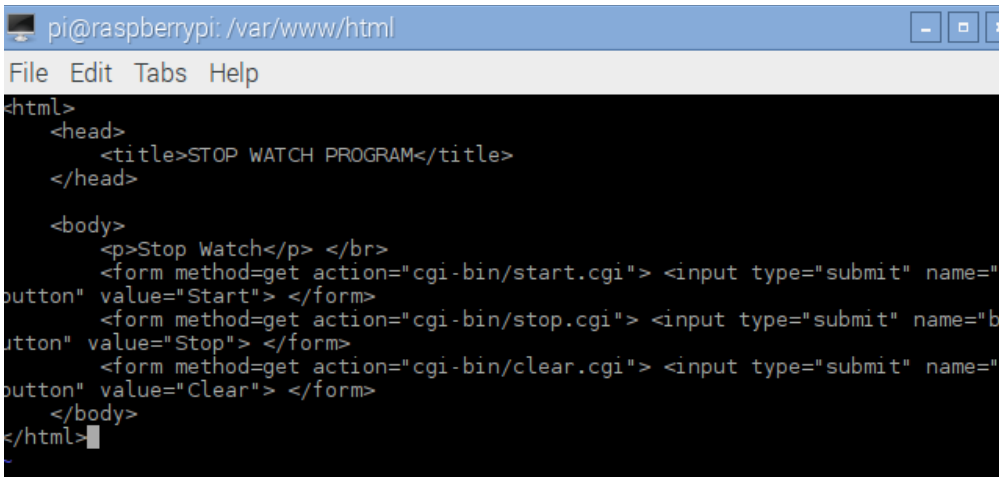
IoT Software 설계 프로젝트

2015104124

진우빈

우선 lab4-4의 예제처럼 html웹페이지를 먼저 만들었습니다.



그리고 start, stop, clear동작을 수행할 c파일을 각각 만들었는데 코드는 다음과 같습니다.

[start.c]

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <fcntl.h>
#include <math.h>
#include <pthread.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <wiringPi.h>
```

```
#define FIFO_FILE     "/tmp/fifo"
#define BUFF_SIZE     1024

// FND - Raspberrypi PIN(S0, S1,  ..., S5)
const int FndSelectPin[6] = { 4, 17, 18, 27, 22, 23 };
// FND LED - Raspberrypi PIN(A, B,  ..., H)
const int FndPin[8] = { 6, 12, 13, 16, 19, 20, 26, 21 };
// FND digits(0 ~ 9) Array
const int FndDigit[10] = { 0x3F, 0x06, 0x5B, 0x4F, 0x66,
                           0x6D, 0x7D, 0x07, 0x7F, 0x67 };
// FND digits with point(0. ~ 9.) Array
const int FndDigit2[10] = { 0xBF, 0x86, 0xDB, 0xCF, 0xE6,
                            0xED, 0xFD, 0x87, 0xFF, 0xE7 };
// timer value
int timer = 0;
bool stop = False;

void Setup() {
      int i;

      if(wiringPiSetupGpio() == -1) {
            exit(-1);
      }

      for(i = 0; i < 6; i++) {
            pinMode(FndSelectPin[i], OUTPUT); // Select PIN OUPUT set
            digitalWrite(FndSelectPin[i], HIGH); // Select PIN OFF
      }

      for(i = 0; i < 8; i++) {
            pinMode(FndPin[i], OUTPUT); // LED PIN OUPUT set
            digitalWrite(FndPin[i], LOW); // LED PIN OFF
      }

      stop = False;
}
```

```
// FND Select function
void FNDSelect(int pos) {
	int i;

	for(i = 0; i < 6; i++) {
		if(i == pos)
			digitalWrite(FndSelectPin[i], LOW);
		else
			digitalWrite(FndSelectPin[i], HIGH);
	}
}

// FND Display function
void FndDisplay(int pos, int num) {
	int i;
	int flag = 0; // FndPin[] ON/OFF
	int shift = 0x01;

	FNDSelect(pos);

	if(pos == 2) {
		for(i = 0; i < 8; i++) {
			flag = (FndDigit2[num] & shift);
			digitalWrite(FndPin[i], flag);
			shift <<= 1;
		}
	}
	else {
		for(i = 0; i < 8; i++) {
			flag = (FndDigit[num] & shift);
			digitalWrite(FndPin[i], flag);
			shift <<= 1;
		}
	}
}

// FND control thread (1/100 sec timer)
```

```
void FndThread(void* arg) {
	while(1) {
		int i;
		int data[10] = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

		for(i = 0; i < 6; i++) {
			FndDisplay(i, data[(timer/(int)pow(10, i))%10]);
			delay(1);
			if((i+timer)%2 == 0)
				delay(1);
		}
		delay(1);

		if(stop) {
			continue;
		}
		timer++;

		if(timer >= 1000000) {
			timer = 0;
		}
	}
}

// IPC thread using FIFO
void IPCThread(void* arg) {
	int fd;
	char buff[BUFF_SIZE];

	if(-1 == mkfifo(FIFO_FILE, 0666)) {
		perror("mkfifo() error!");
		exit(1);
	}

	if (-1 == (fd = open(FIFO_FILE, O_RDWR))) {
		perror("open() error!");
		exit(1);
```

```
	}

	while(1) {
		memset(buff, 0, BUFF_SIZE);
		read(fd, buff, BUFF_SIZE);
		if(buff == "stop") {
			stop = True;
		}
		else if(buff == clear) {
			timer = 0;
		}
		else {
			perror("read buff error!");
			exit(1);
		}
	}
}

int main() {
	printf("Content-type:text/html₩n₩n");
	printf("<html>₩n<head>₩n<title>STOP WATCH START!</title>₩n</head>₩n");
	printf("<body>₩n<p>Start</p>₩n");

	int pid;
	pid = fork();
	if(pid > 0) {
		// parent process
		Setup();
	}
	else if(pid == 0) {
		// child process
		// FND process
		// 0.01 sec stop watch
		pthread_t p_thread[2];
		int thr_id; // thread generation error check
		int status; // value for return value when thread exit
		int a = 1; // thread function value
```

```
            int b = 2; // thread function value
            // thread 1 generation
            thr_id = pthread_create(&p_thread[0], NULL, FndThread, (void *)&a);
            if(thr_id < 0) {
                  perror("thread create error: ");
                  exit(1);
            }
            // thread 2 generation
            thr_id = pthread_create(&p_thread[1], NULL, IPCThread, (void *)&b);
            if(thr_id < 0) {
                  perror("thread create error: ");
                  exit(1);
            }
            // wait thread quit
            pthread_join(p_thread[0], (void **)&status);
            pthread_join(p_thread[1], (void **)&status);
      }
      else if(pid == -1) {
            // error
            perror("fork error : ");
            exit(1);
      }

      printf("</body>₩n</html>");

      return 0;
}
```

Start.cgi에서 자식 프로세스에서 FndProcess를 담당하는데 거기서 1/100초 타이머가 돌아가는 부분과 FIFO로 stop과 clear를 하는 부분을 멀티쓰레드로 나누어 담당하게 하였습니다.

[stop.c]

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/types.h>
```

```c
#include <sys/stat.h>
#include <wiringPi.h>

#define FIFO_FILE     "/tmp/fifo"
#define BUFF_SIZE     1024

int main() {
	int fd;
	char *str = "stop";

	if (-1 == (fd = open(FIFO_FILE, O_WRONLY))) {
		perror("open() error!");
		exit(1);
	}

	write(fd, str, strlen(str));
	close(fd);

	return 0;
}
```

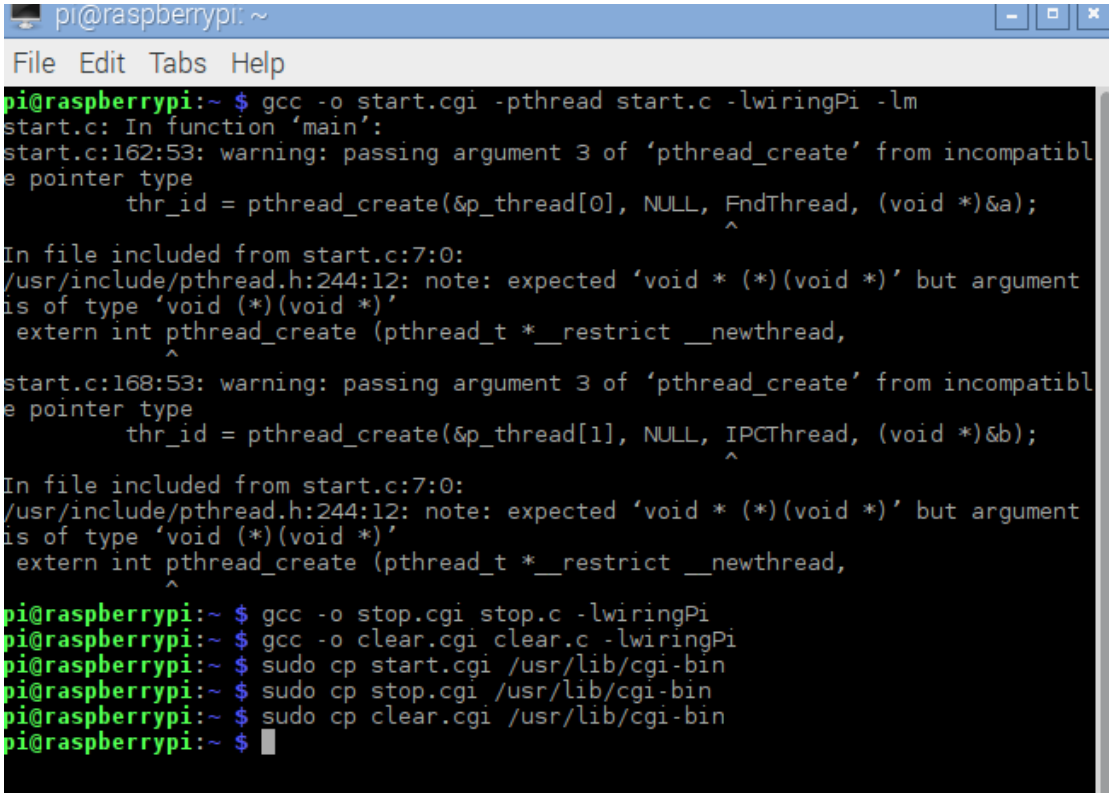
FIFO로 start..cgi의 FndProcess와 통신한다. stop이라는 메시지를 fifo파일로 넘겨줍니다.

[clear.c]

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <wiringPi.h>

#define FIFO_FILE     "/tmp/fifo"
#define BUFF_SIZE     1024

int main() {
```

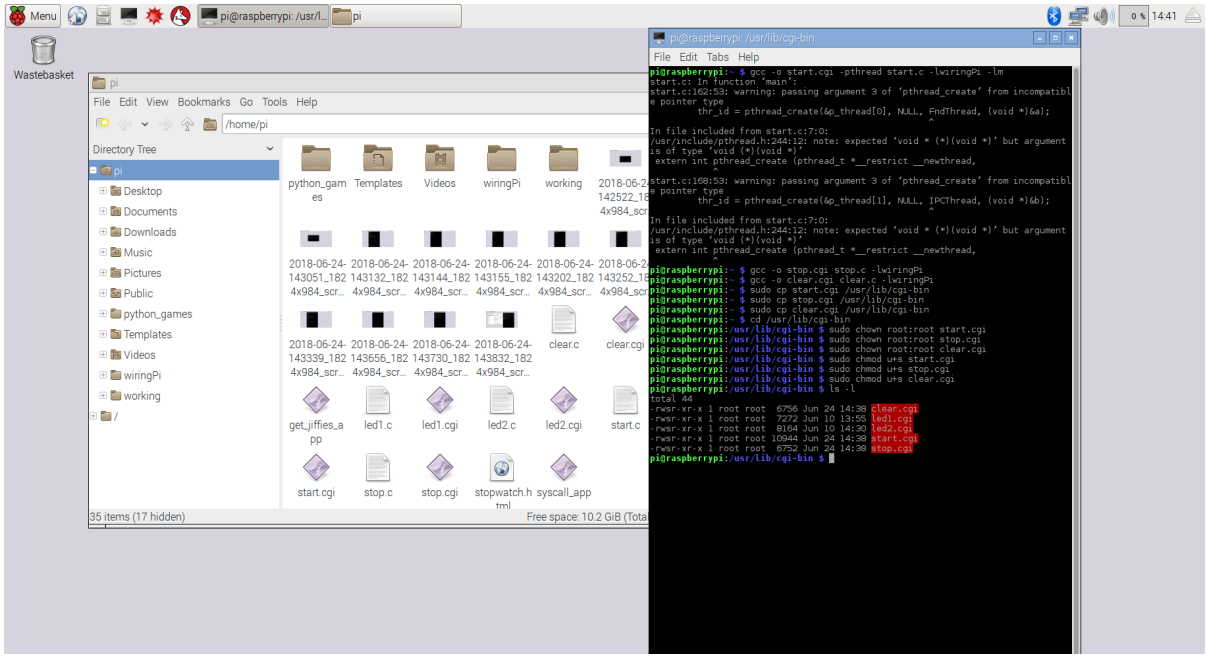
```
    int fd;
    char *str = "clear";

    if (-1 == (fd = open(FIFO_FILE, O_WRONLY))) {
        perror("open() error!");
        exit(1);
    }

    write(fd, str, strlen(str));
    close(fd);

    return 0;
}
```

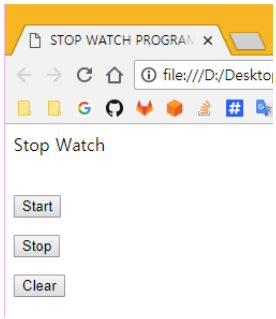
stop.c와 마찬가지로 fifo로 clear라는 메시지를 넘겨줍니다.



위와 같이 각 c파일을 cgi파일로 컴파일 하고 cgi-bin폴더로 복사한 후

아래와 같이 권한을 바꿔준 뒤 같은 네트워크의 다른 컴퓨터에서 웹페이지를 실행해 보았습니다.





실행 결과 FND 의 동작을 확인할 수 있었습니다.

폴더에 해당 코드를 첨부하였습니다.