

# lot 프로그래밍 프로젝트 2팀

장창용 최다원 박상운 허지웅



# 목차



1. 프로젝트 소개
2. 기능구현
  - Dot-Matrix
  - Tact Switch
  - CLCD
  - FND
3. 차별점
4. 참고자료



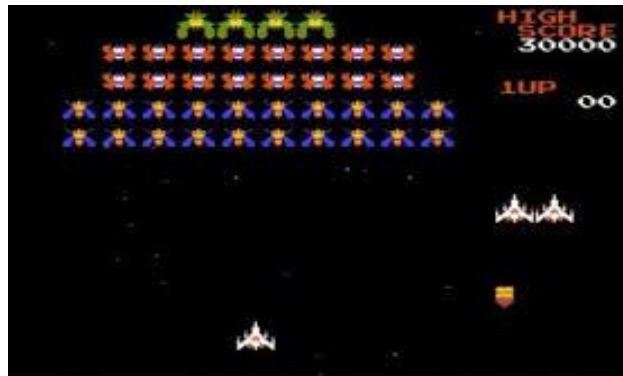
# 1. 프로젝트 소개

# 1. 프로젝트 소개

## 갤러그 게임

위에서 내려오는 적을 플레이어가 좌우로 이동하여 슈팅을 통하여 적을 없애는 게임  
적을 맞추면 점수 +1점 적과 닿으면 게임 종료

시연 영상 : <https://youtu.be/sj5oAKozdKo>





## 2. 기능 구현

## 2.1 Dot-Matrix 플레이어/적

-for문을 사용하여 초기 상위 두줄로 적을 생성하고 적이 한 줄씩 밑으로 이동하도록 생성

-적과 플레이어가 닿으면 게임 종료, 플레이어가 살아있다면 5초마다 적이 내려오도록 설정

```
//한 줄씩 내려오는 함수 - false: gameover, true: keepgoing
bool enemyGoingDown()
{
```

```
    //내려올 시 플레이어 함선 앞부분에 닿으면 게임오버
    for (int i = 0; i < 8; i++)
    {
        if (matrix[5][i])
            return false;
    }
}
```

```
//한줄씩 밑으로 이동
for (int i = 5; i > 0; i--)
{
    for (int j = 0; j < 8; j++)
    {
        setPoint(i, j, matrix[i - 1][j]);
    }
}
```

```
//디버깅용 프린트
for (int i = 0; i < 8; i++)
{
    for (int j = 0; j < 8; j++)
    {
        printf(matrix[i][j] ? "1" : "0");
    }
    printf("\n");
}
```

```
printf("\n");
```

```
//가장 윗줄 적으로 채우기
for (int i = 0; i < 8; i++)
    setPoint(0, i, 1);
```

```
return true;
}
```

```
//플레이어 사망 전까지 루프
```

```
while (isalive)
{
    //총알
    bulletLogic();

    //출력- dotmatrix, fnd
    drawToMatrix(TIME_QUANTUM * 25); // 25/600초
    drawFND(TIME_QUANTUM * 5, level); // 5/600초
}
```

```
//tact switch 명령 가져오기
command = getButtonTACT();
```

```
if (command > -1 && command < 3)
{
    printf("command - %d \n", command);
    ship.controll(command);
    setPlayerPos();
}
```

```
timer++;
// 위의 출력하는 것들 usleep초 계산
// 30/600 이니 20번 수행하면 1초가 지나감
if (!(timer % 20))
```

```
{
    timer = 0;
    //5초마다 내려오는 적들
    downCounter++;
    printf("downCounter - %d\n", downCounter);
}
```

```
if (downCounter == 5)
{
    //플레이어가 살아있으면 적이 내려옴
    isalive = enemyGoingDown();
    downCounter = 0;
    level++;
}
```

## 2.2 Tact-Switch 이동 / 옵션

### 플레이어 이동

-2번 입력시 공격, 4번 입력시 왼쪽 이동  
6번 입력시 오른쪽 이동, 5번 입력시 공격

### 시작 및 확인/재시작

-시작시 3번키 입력받아 시작

```
//TactSwitch
int getButtonTACT()
{
    unsigned char b;
    int tactswFd = -1;
    tactswFd = open(tactswDev, O_RDONLY);
    read(tactswFd, &b, sizeof(b));
    close(tactswFd);
    switch (b)
    {
        case 2: //공격
            return 0;
        case 4: //왼쪽 이동
            return 1;
        case 6: //오른쪽 이동
            return 2;
        case 5: //확인 버튼
            return 3;

        default:
            return -1;
    }
}

while (true)
{
    //게임 시작전 확인버튼
    beforeGameCLCD();
    while (getButtonTACT() != 3)
    {
        printf("Press StartButton \n");
    }

    init();

    //한번더 확인
    while (getButtonTACT() != 3)
    {
        printf("one more\n");
        drawToMatrix(TIME_QUANTUM * 20);
    }
}
```

## 2.3 CLCD 안내문구 / 점수

### 점수 반환 함수

-sprintf를 사용해 점수 문자열 변환  
지정된 길이 space만큼 앞에 0을 채워  
"Score : 0000" 형태의 scoreSpacer 반환

### CLCD 출력 함수

-안내 문구(시작/최고 점수.현 점수/종료)

```
//CLCD
int clcd_fd;
// CharacterLCD에 출력
void printCLCD(string S)
{
    char* cstr = new char[S.length() + 1];
    strcpy(cstr, S.c_str());
    clcd_fd = open(CLCD, O_WRONLY);
    write(clcd_fd, cstr, 32);
    close(clcd_fd);
}

string scoreSpacer(int I, int space)
{
    //c++11 to_string 이 사용이 안되어 char 배열 에서 string으로 변환
    char s[10];
    sprintf(s, "%d", I);
    string STR(s);

    for (int i = STR.length(); i < space; i++)
    {
        STR = '0' + STR;
    }
    return STR;
}

void beforeGameCLCD()
{
    string s1 = "      Galaga      ";
    string s2 = "    Press Start    ";
    printCLCD(s1 + s2);
}

void gamingCLCD(int score, int highScore)
{
    string s1 = "Score   : " + scoreSpacer(score, 4) + " ";
    string s2 = "HighScore: " + scoreSpacer(highScore, 4) + " ";
    printCLCD(s1 + s2);
    void gamingCLCD(int score, int highScore)
    {
        string s1 = "Score   : " + scoreSpacer(score, 4) + " ";
        string s2 = "HighScore: " + scoreSpacer(highScore, 4) + " ";
        printCLCD(s1 + s2);
    }

    void gameOverCLCD(int score)
    {
        string s1 = "Score   : " + scoreSpacer(score, 4) + " ";
        string s2 = "    Game Over    ";
        printCLCD(s1 + s2);
    }
}
```



## 2.4 FND

-FND를 제어하고 레벨 값을 표시하는 기능  
drawFND() 함수를 호출하여 지정된 시간  
동안 레벨 값을 FND에 출력  
asc\_to\_fnd() 함수는 숫자를 FND에 표시하기  
위한 비트 패턴으로 변환


-4자리로 표시되며, 각 자릿수는 0부터  
9까지의 숫자 또는 공백으로 표현

```
//FND
int fnd_fd = -1;
unsigned char asc_to_fnd(int n)
{
    //0 ~ 9까지의 숫자들
    unsigned char FND_BITS[11] = { 0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xD8, 0x80, 0x98, 0xFF };
    return n >= 0 && n < 10 ? FND_BITS[n] : FND_BITS[10];
}

// 지정된 시간동안 FND에 출력
void drawFND(int microSec, int level)
{
    fnd_fd = open(FND_DEV, O_WRONLY);
    unsigned char Hex_Code[4];
    memset(Hex_Code, 0x00, sizeof(Hex_Code));

    int TargetNum = level;
    int i;
    for (i = 1; i < 4 + 1; i++)
    {
        int Num = TargetNum / pow(10, 4 - i);
        TargetNum -= Num * pow(10, 4 - i);
        Hex_Code[i - 1] = asc_to_fnd(Num);
    }
    write(fnd_fd, Hex_Code, 4);
    usleep(microSec);
    close(fnd_fd);
}
```


# 차별점



## 기존 갤러그 게임과 다름

- 기존 갤러그 게임 방식 : 적들이 위에서 내려오지 않고 나뼌의 특색있는 공격을 통하여 플레이어에게 위협
- 위에서 한 칸씩 내려오는 스페이스 인베이더의 기능을 가져와 기존과는 다른 게임 구현

## 참고 자료



- <https://github.com/B31l/Smart4412Linux?tab=readme-ov-file#%EC%BD%94%EB%93%9C>
- <https://github.com/jinwoo1225/SnakeGameWithSmart4412>
- [https://github.com/JoHyeonGyeong/iot\\_indianPoker](https://github.com/JoHyeonGyeong/iot_indianPoker)



감사합니다