

## Systemy Wbudowane

### GameDev (Wii)

Prowadzący:

Jakub Szygula: [jakub.szygula@polsl.pl](mailto:jakub.szygula@polsl.pl)

Dariusz Marek: [dariusz.marek@polsl.pl](mailto:dariusz.marek@polsl.pl)

#### 1. Wstęp:

W ramach ćwiczenia laboratoryjnego przygotowane zostaną aplikacje na konsolę Nintendo Wii, w oparciu o środowisko DevkitPro. Udostępnia ono narzędzia do projektowania homebrew (nieoficjalne środowisko programistyczne, z zakazem czerpania zysków z wytworzonego oprogramowania), które są dedykowane na wiele popularnych konsoli, w tym Game Boy Advance, DS i GP32 (devkitARM); PlayStation Portable (PSP) (devkitPSP); oraz GameCube i Wii (devkitPPC). Narzędzia te są dostępne dla systemów Windows, Mac OSX i Linux.

Techniki programistyczne które będą wykorzystywane w trakcie laboratorium oparte są o często stosowane API **OpenGL** (ang. **Open** Graphics Library).

Możliwa jest obsługa ruchomego zewnętrznego kontrolera Wii Mote:



Wymagane oprogramowanie:

DevkitPro Updater v3.0.3:

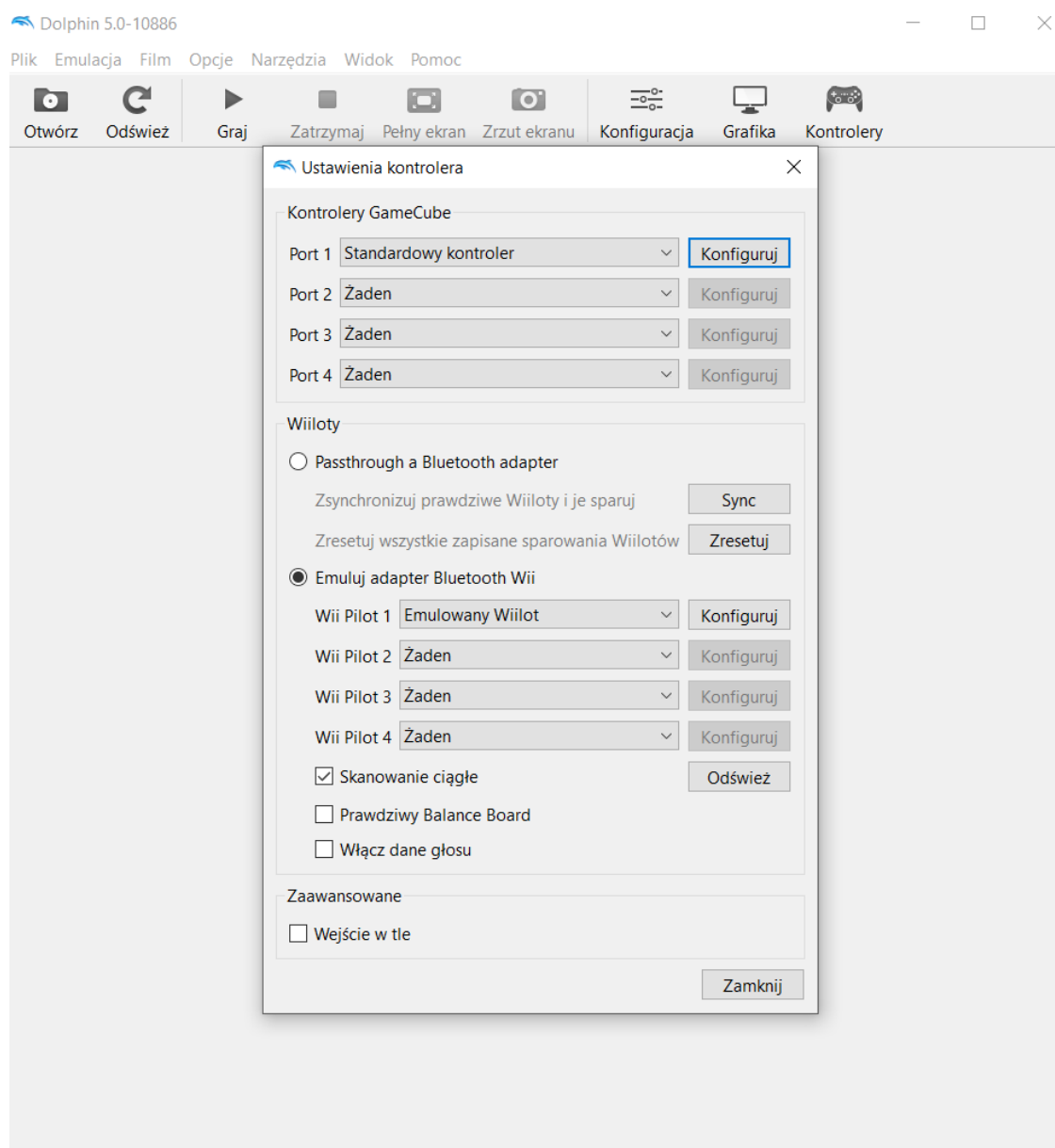
[https://github.com/devkitPro/installer/releases?fbclid=IwAR07JOx67NPxfCQd3i953nnvUoTi2Y3Vr\\_xV\\_TzrGNKdQxnl7TAfBc6LPqY](https://github.com/devkitPro/installer/releases?fbclid=IwAR07JOx67NPxfCQd3i953nnvUoTi2Y3Vr_xV_TzrGNKdQxnl7TAfBc6LPqY)

Emulator Dolphin (emulator konsol firmy Nintendo, w tym GameCube oraz Wii):

<https://pl.dolphin-emu.org/>

Emulator wystarczy pobrać i rozpakować, a następnie poprawnie ustawić:

Ustawienia sterowania w emulatorze:



## Ustawienia Wii Pilota:

Wii Pilot 1

Urządzenie

DInput/0/Keyboard Mouse

Odśwież

Zresetuj

Domyślne

Wyczyść

Profil

Wczytaj

Zapisz

Usuń

Ogóły i opcje

Motion Controls

Rozszerzenie

Przyciski

A

Click 0

B

Click 1

1

1

2

2

-

Q

+

E

HOME

!LMENU & ...

D-Pad

Góra

UP

Dół

DOWN

Lewo

LEFT

Prawo

RIGHT

Skróty klawiszowe

Sideways Toggle

Upright Toggle

Sideways Hold

Upright Hold

Rozszerzenie

Nunchuk

Attach MotionPlus

Wibracje

Motor

Opcje

Speaker Pan

0,00 %

Bateria

95,00 %

Wiiot trzymany pionowo

Wiiot trzymany poziomo

Zamknij

Wii Pilot 1

Urządzenie

DInput/0/Keyboard Mouse

Odśwież

Zresetuj

Domyślne

Wyczyść

Profil

Wczytaj

Zapisz

Usuń

Ogóły i opcje

Motion Controls

Rozszerzenie

Wstrząs

X

Click 2

Y

Click 2

Z

Click 2

Dead Zone

0,00 %

Intensity

10,00 cm

Frequency

6,00 Hz

Point

Calibrate

Góra

Cursor Y-

Dół

Cursor Y+

Lewo

Cursor X-

Prawo

Cursor X+

Ukryj

Wyśrodkuj

Relative Input Hold

Dead Zone

0,00 %

Vertical Offset

10,00 cm

Total Yaw

15,00 °

Total Pitch

15,00 °

Wejście relatywne

Auto-Hide

Przechylenie

Calibrate

W przód

W tył

Lewo

Prawo

Zmiennik

Dead Zone

0,00 %

Kąt

90,00 °

Zamach

Calibrate

Góra

Dół

Lewo

Prawo

W przód

W tył

Dead Zone

0,00 %

Distance

50,00 cm

Speed

16,00 m/s

Return Speed

2,00 m/s

Kąt

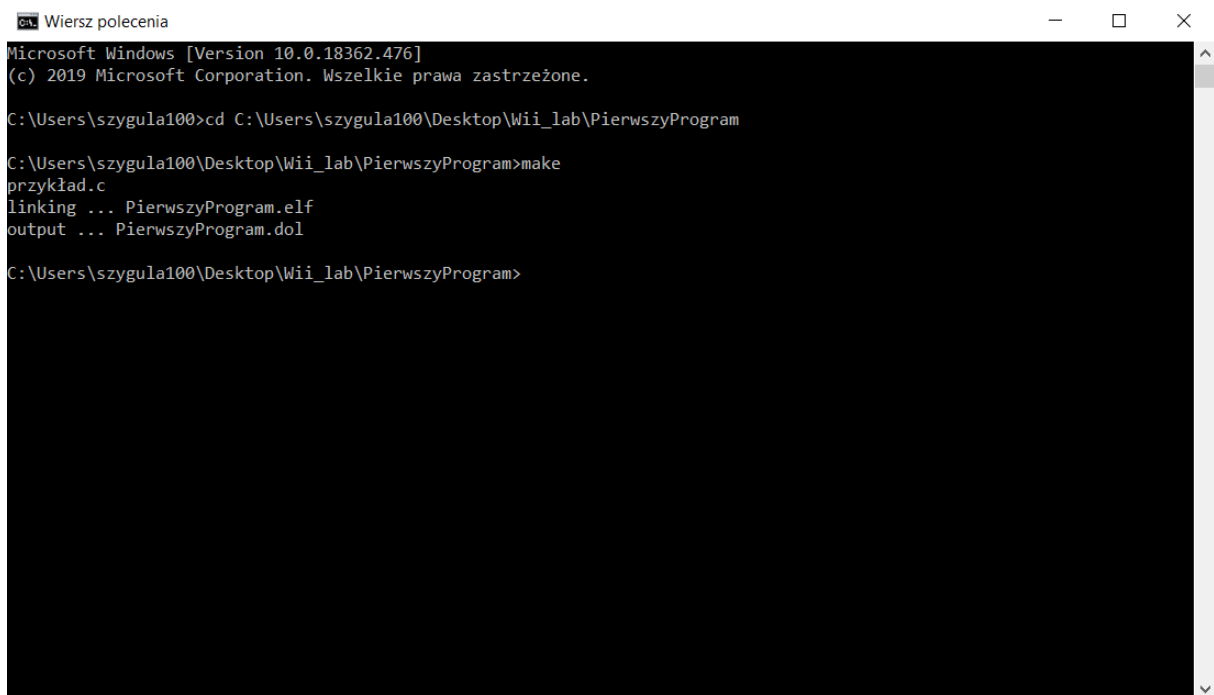
90,00 °

Zamknij

## 2. Tworzenie pierwszej aplikacji na Wii

W folderze PierwszyProgram, znajduje się kod pozwalający na stworzenie pierwszej aplikacji na konsolę Wii, mający za zadanie zainicjalizować wyświetlanie ekranu startowego, a także pobierać współrzędne kontrolera ruchowego (Wii Remote), by następnie wyświetlać jego punkt położenia.

By poprawnie zbudować aplikację (po wcześniejszej instalacji Devkit Pro), należy w konsoli przejść do folderu zawierającego PierwszyProgram, a następnie wykonać make.



```
Wiersz polecenia
Microsoft Windows [Version 10.0.18362.476]
(c) 2019 Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Users\szygula100>cd C:\Users\szygula100\Desktop\Wii_lab\PierwszyProgram

C:\Users\szygula100\Desktop\Wii_lab\PierwszyProgram>make
przykład.c
linking ... PierwszyProgram.elf
output ... PierwszyProgram.dol

C:\Users\szygula100\Desktop\Wii_lab\PierwszyProgram>
```

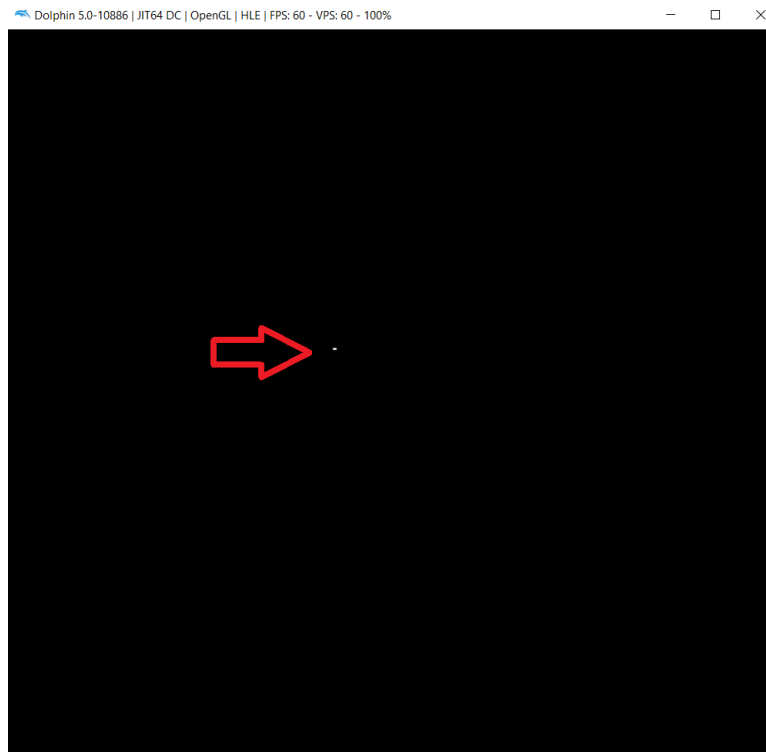
Z perspektywy przygotowania aplikacji na konsolę Nintendo Wii, w pliku Makefile najważniejsze jest zdefiniowanie obsługi kontrolerów:

```
include $(DEVKITPPC)/wii_rules
```

Po prawidłowym wykonaniu make, zbudowany plik .dol należy otworzyć w emulatorze Dolphin.

## 3. Opis pierwszej aplikacji

Przykład ma na celu stworzenie pointer'a, przedstawiającego położenie kontrolera Wii (lub myszki – w przypadku emulatora).



Najważniejsze funkcje zmienne i struktury:

```
GXRModeObj *rmode;
```

Przechowuje wysokość i szerokość tworzonego okna

Link do dokumentacji:

[https://libogc.devkitpro.org/gx\\_struct\\_8h.html#a232a76ec57f12bf0e45abad393c36698](https://libogc.devkitpro.org/gx_struct_8h.html#a232a76ec57f12bf0e45abad393c36698)

```
WPAD_Init();
```

Inicjalizuje połączenie z kontrolerami

```
WPAD_SetVRes(WPAD_CHAN_ALL, rmode->fbWidth, rmode->xfbHeight);
```

Inicjalizacja wszystkich dostępnych kontrolerów (WPAD\_CHAN\_0 inicjalizowałoby połączenie wyłącznie z pierwszym kontrolerem) oraz ustawienie szerokości i wysokości okna.

```
ir_t Ir
```

Zmienna przechowująca dane kontrolera (m.in. współrzędne jego położenia)

```
WPAD_ScanPads();
```

Rozpoczęcie skanowania położenia kontrolerów

```
if (WPAD_ButtonsDown(0) & WPAD_BUTTON_HOME)
{
    exit(0);
}
```

Przykład obsługi przycisku kontrolera, a także najważniejszy fragment programu – pozwalający na przerwanie pracy aplikacji w sposób inny niż wyłączenie całej konsoli 😊

Przyciski Wii kontrolera:

```
#define WPAD_BUTTON_2
#define WPAD_BUTTON_1
#define WPAD_BUTTON_B
#define WPAD_BUTTON_A
#define WPAD_BUTTON_MINUS
#define WPAD_BUTTON_HOME
#define WPAD_BUTTON_LEFT
#define WPAD_BUTTON_RIGHT
#define WPAD_BUTTON_DOWN
#define WPAD_BUTTON_UP
#define WPAD_BUTTON_PLUS
```

Przyjmuje się, że podstawowe przyciski kontrolera to A i B (z przodu i z tyłu pilota).

#### **Dokumentacja :**

#include <wiiuse/wpad.h>

<https://github.com/devkitPro/libogc/blob/master/gc/wiiuse/wpad.h>

Ponadto, w folderze instalacyjnym DevkitPro dostępnych jest wiele innych przykładów implementacji na Nintendo Wii:

Ścieżka: devkitPro\examples\wii

## **4. Zadania do realizacji**

**Uwaga: Zadanie 4.5 (tekstury) jest w pełni odrębne od pozostałych, można je zrealizować bez wykonania wcześniejszych punktów.**

**Do zadania 3 wykorzystaj dostarczony template 'zad.3' (OpenGL z kolorami)**

**Do zadania 4 wykorzystaj ponownie template 'zad.3' (OpenGL z kolorami)**

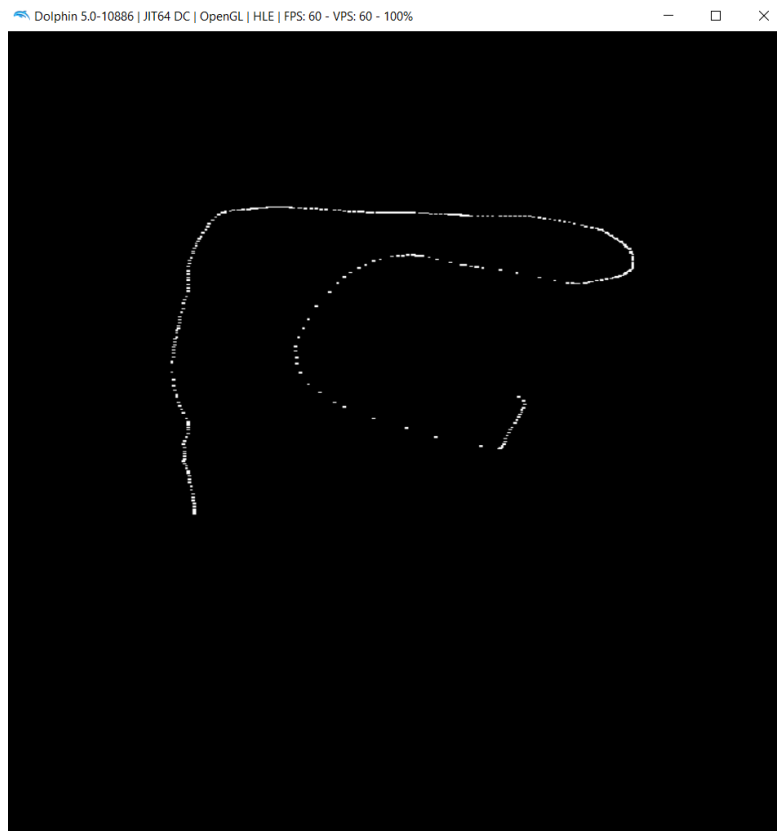
**Do zadania 5 wykorzystaj template 'zad.5' (animacje)**

- 1) Wykorzystaj przygotowany szablon PierwszyProgram i napisz program mapujący ścieżkę poruszania się pointera na ekranie (zgodnie z poniższą przykładową ilustracją):

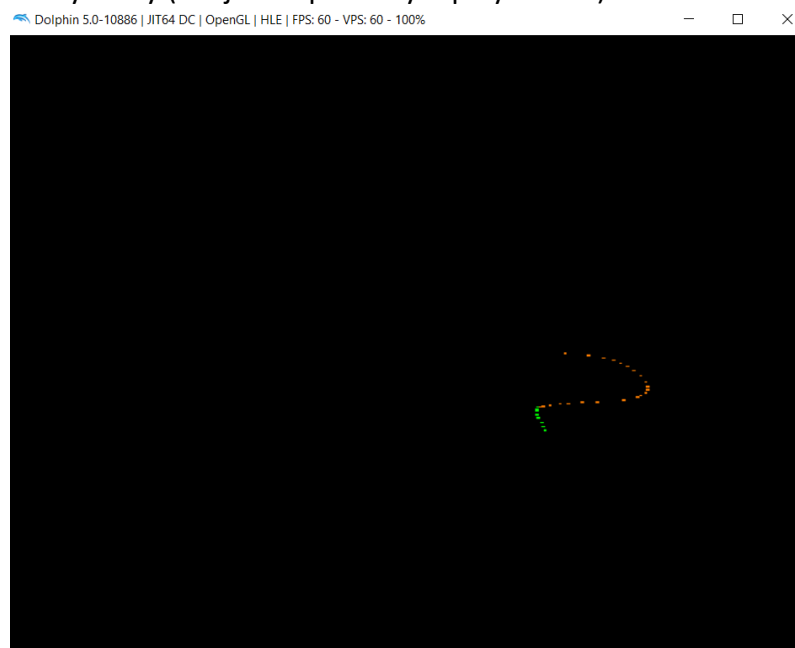
Podpowiedź: Możesz wykorzystać

```
static bool *map;
```

do oznaczania wyświetlanych punktów.



- 2) Napisz aplikację, w której narysowane punkty wyświetlają się przez 0,5 sekundy (zakładając 60 FPS), a następnie znikają.
- 3) Rozwiń funkcjonalność zadania drugiego, poprzez dodanie kolorów. Rysowane punkty nie powinny być białe, natomiast kiedy wciśnięty jest przycisk – kolor powinien być inny (tak jak na poniższym przykładzie):



Podpowiedź:

Możesz wykorzystać poniższą tablicę do określania kolorów:

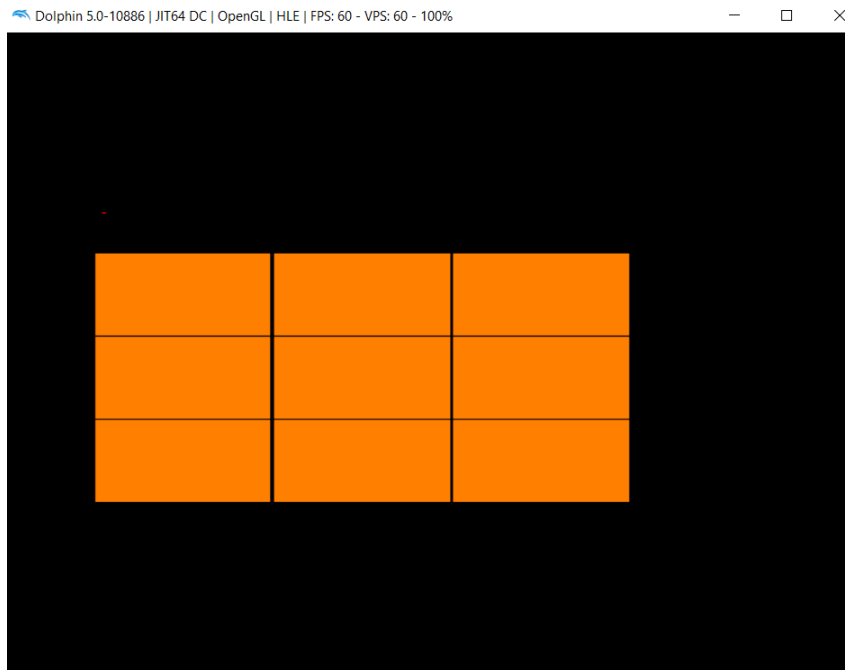
```
static f32 BoxColors[5][3] = {  
    {1.0f, 0.0f, 0.0f}, // Czerwony  
    {1.0f, 0.5f, 0.0f}, // Pomarańczowy  
    {1.0f, 1.0f, 0.0f}, // Żółty  
    {0.0f, 1.0f, 0.0f}, // Zielony  
    {0.0f, 1.0f, 1.0f}  // Niebieski
```

Sposób odwołania do przygotowanych kolorów wewnątrz funkcji drawQuad:

```
void drawQuad(int x, int y, int width, int height, f32 color[3])  
{  
    GX_Begin(GX_QUADS, GX_VTXFMT0, 4); // Rozpoczęcie rysowania kwadratu  
    GX_Position3f32(x, y, 0.0f); // Pozycja lewego górnego narożnika  
    kwadratu na scenie  
    GX_Color3f32(color[0], color[1], color[2]);  
    GX_Position3f32(x + width - 1, y, 0.0f); // Pozycja prawego górnego  
    narożnika kwadratu na scenie  
    GX_Color3f32(color[0], color[1], color[2]);  
    GX_Position3f32(x + width - 1, y + height - 1, 0.0f); // Prawy dół  
    GX_Color3f32(color[0], color[1], color[2]);  
    GX_Position3f32(x, y + height - 1, 0.0f); // Lewy dół  
    GX_Color3f32(color[0], color[1], color[2]);  
    GX_End(); // Zakończenie rysowania kwadratu  
}
```

- 4) Stwórz szyfrator graficzny, znany m.in. z blokad telefonicznych.  
Wykorzystaj funkcjonalności pozwalające na obsługę kontrolera oraz rysowanie kolorowych punktów i kwadratów.
  - a) W pierwszej kolejności wyświetl kolorowe bloki na ekranie.





- b) Do momentu pierwszego zatwierdzenia przyciskiem, pozwól na wprowadzenie odpowiedniego wzoru.



- c) Wprowadź szyfr i poprawne kroki oznaczaj kolorem zielonym, a błędne czerwonym. Ponowne naciśnięcie przycisku ma pozwolić na następną próbę odgadnięcia zapisanego szyfru.



Podpowiedź:

Przygotuj odpowiednią strukturę, która umożliwi oznaczanie kolorowych bloków ('id', pozwoli to na zapisywanie kolejności wybieranych bloków) oraz przechowywania ich wymiarów i współrzędnych.

- 5) Przeanalizuj przykład umożliwiający wyświetlanie tekstury. Rozwiń funkcjonalność aplikacji zgodnie z wytycznymi opisanymi w komentarzach 'zad5.c'.

Wczytanie tekstury wymaga przygotowania pliku ze ścieżką zawierającą jej nazwę oraz późniejszy sposób odwołania (jak w dołączonym przykładzie).

```
<filepath="ballsprites.png" id="ballsprites" colfmt=6 />
```

## 5. Zaliczenie laboratorium i sprawozdanie

W celu zaliczenia laboratorium należy pokazać działające zadania, przetestować przygotowane oprogramowanie na konsoli Nintendo Wii 😊 oraz przygotować sprawozdanie (kody źródłowe oraz opis wykonanych zadań).