東海大学紀要情報通信学部
Vol.8,No2,2015,pp.21-29

論文

ゲーム開発によるモチベーションを維持した組込みシステム導入教育

古川 拓実[*1]，清水 尚彦[*2]

Enhancing Motivation in Learning of Embedded System with Game Development

by

Takumi FURUKAWA[*1] and Naohiko SHIMIZU[*2]

(received on Sep. 25, 2015 & accepted on Jan. 14, 2016)

あらまし

プログラミング未経験者から初学者を対象とする組込みシステム教育の導入としてゲーム開発を題材とした教材を開発した．ゲーム開発を題材とすることで学習者のモチベーションを維持しつつハードウェアを意識させることのできる教育を行う．我々はゲーム開発を組込みシステム教育へ利用できるライブラリを開発し，それらを用いた教材も開発した．教材を用いた授業を企画し，プログラミング初学者へ実施したところ，80%以上の受講者がハードウェアの操作を実感でき，70%が勉強する意欲を増大させた．本論文では，教材の開発過程と授業実施までのプロセスの報告と授業結果の考察を行う．

Abstract

For learning embedded systems, we have developed a teaching material with game development. The target of learning is beginner programmers. The game development has the effect of improvements in user motivation. We performed a lesson using this teaching material. As the result of lesson, about 80% of the students could feel the operation of hardware and 70% of the students increased willingness to learn. In this paper, we report the development process of the teaching material and result of lesson.

キーワード: 組込みシステム，教育，ゲーム開発
Keywords: embedded system , education, game development

1. はじめに

組込みシステムはロボット，家電などの情報機器や自動車など様々な分野で用いられている．しかし，組込みシステムの開発スキルを持った技術者が不足しており，ソフトウェアだけでなくハードウェアの知識を持った技術者の育成が必要不可欠となっている．

また，教育機関ではプログラミング教育にゲーム開発を取り入れ，学習者のモチベーションを維持した教育が行われている[1)~4)]．これらの学習は対象をプログラミング初学者やプログラミング未経験者としており，ソフトウェアやWebを用いて構文や動作を学習している．組込みシステム技術者はこれらのプログラミングスキルに加えて，ハードウェアを意識できるスキルが求められる．経済産業省の「組込みスキル標準」（ETSS）[5)]では，組込みソフトウェア開発は使用される技術が多岐に渡ることから技術者へ要求されるスキルが増大する問題を上げている．ソフトウェアだけではなく多様なスキルの習得を必要とするため，学習コストが高くソフトウェアのみの学習に比べて長期になる．ETSSでは組込みシステム開発未経験者向け教育プログラム[6)]にてC言語を中心にメモリ，I/Oなどのハードウェアを習得し組込みに必要なスキルのギャップの解消を上げている．本研究ではゲーム開発を組込みソフトウェア教育へ取り入れ，学習者のモチベーションを維持しつつハードウェアを意識できる教材を開発することで組込み分野への人材シフトを容易に行える教育を目的とする．

本研究で行う授業をETSSのスキル標準で表記したものをTable1に示す．本授業は組込みシステムの分野において未学習者や初学者に基礎的なスキルであるユーザインターフェースやプロセッサ，デバイスの制御に関する知識を身につけてもらうことが目的である．対象例として高校生または情報分野の大学生が当てはまる．システムを動作させるために，最低限プログラミングスキルを必要とする．しかし，学習状況によってプログラミングスキルに差が出てしまう．自身でプログラミングできる者(Level2)だけでなく，プログラム未学習者(unlearned)や初学者(Level1)も組込みシステムの学習を行える授業のために,我々はサンプルを組み合わせることで学習状況

*1 情報通信学研究科　情報通信学専攻　修士課程
Graduate School of Information and Telecommunication Engineering, Couse of Information and Telecommunication Engineering, Master's program
*2 情報通信学部　組込みソフトウェア工学科　教授
School of Information and Telecommunication Engineering, Department of Embedded Technology, Professor

Table1 ETSS skill category

Skill Category	Class	Education Item	Target	Goal
Technical Elements	Multimedia	Sound/Video/Image	Unlearned	Level1
	User Interface	Button	Unlearned	Level1
		Display/Sound	Unlearned	Level1
	Storage	Memmory	Unlearned	Level1
	Processor	Multi Processor	Unlearned	Level1
Development Technology	Software code development and Test	Development Environment	Unlearned	Level1
		Emulation	Unlearned	Level1
		C Language	Unlearned	Level1
		C Language	Level1	Level2

に臨機応変に対応できる教材を開発した．ゲーム開発に適した形に抽象化したライブラリを開発し，それを用いたゲーム開発を行うことでプログラミングの学習状況を問わず組込みシステム開発を実感できるプログラミング学習が行える．

本論文では，ハードウェアを容易に制御できるライブラリの作成とそれを用いたゲーム教材の開発，開発した教材を用いた授業について報告する．

2. 教材

ハードウェアを用いたゲームの対象として任天堂社のファミリーコンピュータ(ファミコン(英名NES))[7]を選択した．既存のゲーム機で動作可能なゲームソフトウェアを開発することで，作成したソフトウェアでハードウェアを動かす実感を高め，さらなるモチベーションの向上となる．

ファミコンのハードウェアを Fig.1 に示す．ETSSのスキル要素とファミコンのハードウェア操作による学習可能な要素の対応表を Table2 に示す．組込みシステムは通常のパソコンと違い，プロセッサを複数持ち，プロセッサ間の制御を行うソフトウェアを必要とすることが多い．ファミコンでは，CPU の他に，サウンド生成を担う Audio Processing Unit (APU) と画面描画全般を受け持つ Picture Processing Unit (PPU) の 2 つのプロセッサを持つ．そのため，ファミコンゲーム開発ではゲームの動作の他にレジスタ書き込みを用いて APU や PPU を操作し，サウンドや映像を生成する必要がある．外部からの信号入力として 8 ボタンのコントローラ(PAD)を接続している．ボタン押下データの取得タイミングやチャタリングなど入力デバイスの制御に関する学習を行うことができる．外部への出力として PPU から映像出力，APU からサウンド出力を行っている．映像出力は画面の水平同期や垂直同期を考慮しなければならないため，画面の周波数の理解や仕様に合わせた制御の学習ができる．その他にも搭載メモリの上限や CPU の動作周波数など組込みシステムの導入教育に必要な要素を備えている．このことから我々は教材とするゲーム機としてファミコンを選択した．

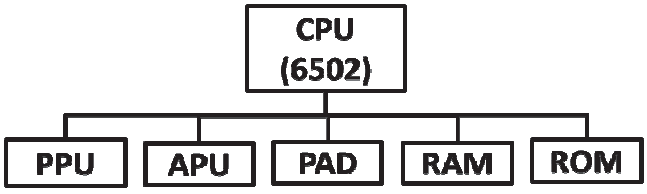

Fig.1 Component of NES Hardware
CPU controls the other modules.

Table2 Correspondence table

Skill	Factors
Multimedia	Sound/Video generation
User Interface	Input by the PAD
	Output to the display
Storage	RAM・ROM
Processor	CPU・APU・PPU
Software code development and Test	Test with emulator

ファミコンは CPU に 6502 プロセッサを搭載しており，cc65 コンパイラ[8]を用いて C 言語をコンパイルすることができる．しかし，ファミコンはメモリマップド I/O でデバイスの操作を行っているため，ポインタやメモリに関する知識を必要とする．このままでは教育対象者が C 言語のポインタやメモリ関連の学習を終えている者になってしまう．我々はデバイスの操作ごとに処理をまとめたライブラリを開発し，それを用いたゲーム開発を行うことでポインタに関して未習得または学習途中の者でも組込みシステムの学習を行うことができるようにした．さらに，ハードウェアの動作を一つ一つ理解できるサンプルを用意し，サンプル同士を組み合わせてゲームが完

Table3 How to use each skill

Programming skill of learner	To do
Unskilled/ Beginner(Branch/Loop)	Rewrite of sample
Array/Function	Combine of sample / New development
Pointer	New development / Rewrite of library

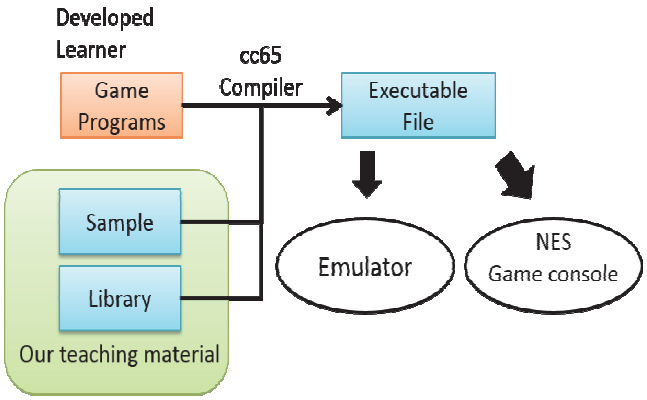

Fig.2 Development Flow of our teaching material

成できるようにすることで C 言語初学者や C 言語学習を行っていない者でもハードウェアの操作を体験できるようにした．Table3 にスキル別の教材利用方法を示す．教材を用いたゲーム開発フローを Fig.2 に示す．サンプルプログラムや学習者が作成したプログラムをライブラリと共に cc65 コンパイラで実行ファイルを生成する．ファミコンエミュレータ上でゲームを実行し，プログラムの動作確認を行う．完成したプログラムは ROM へ書き込むことにより，ファミコン実機で動作することが可能な形式である．ハードウェア仕様やライブラリとサンプルプログラムの解説を資料として配布し，ステップごとの動作の確認や機能の追加を学習者が行えるようにする．組込みシステムではデバイスの仕様や API の資料からデータを読み取りソフトウェアを設計することがある．ファミコンのハードウェアやゲーム開発環境の資料からプログラムの読み取り，設計を行うことで実際の組込みシステム開発と同じ状況で開発を行うことができる．

2.1 ライブラリ

ファミコンのゲームはポインタを複雑に用いてプログラミングする．中でも，画面操作，音生成やボタン入力は特に複雑なものとなる．我々は学習者の C 言語のスキルレベルによらずハードウェアの操作の学習を行うために，ポインタ操作部分を処理単位でまとめたライブラリを開発した．開発したライブラリは大きく分けて映像操作関連をまとめた PPU，サウンド関連をまとめた APU，PAD のからの入力をまとめた PAD の 3 つである．

(1) PPU

ファミコンの映像はバックグラウンド(BG)とスプライト(SP)の 2 枚の画面を合成して作成される．画面サイズは横 256 ピクセル，縦 240 ピクセルである．PPU は 1/60 秒で一枚の画面を出力し，垂直同期を取る．一枚の画面描画が終わってから次の画面の描画が始まるまでの区間を Vblank といい，この区間に CPU は PPU へ BG や SP の画像データや色情報を書き込む．我々は PPU へのデータ書き込みや読み込みにポインタを用いている部分をまとめ，PPU の操作という単位で扱う．これによりポインタを用いずに画面の操作をできるようにした．PPU の操作をまとめ，作成したメソッドを Table4 に示す．作成したメソッドを PPU 全体の処理，BG の処理，SP の処理と分け，よりわかりやすいライブラリを開発した．PPU を使った基本的な処理のフローを Fig.3 に示す．waitvblank メソッドで vblank を待ち，ppu_off メソッドで描画を止めてから PPU へデータを書き込む．

Table4 PPU library method

Method		Detail
Global method	waitvblank	Wait for next Vblank
	ppu_on	Drawing start
	ppu_off	Drawing stop
	palette_change_all	Writing of color information
	palette_change_BG	
	palette_change_SP	
Back glould method	getBG	Get the image number of BG tile
	setBG	Change the image number of BG tile
	set_scroll	BG scrolling
	clear_BG	Reset BG data in PPU
Sprite method	update_sprite	Writing of sprite data in SP_DATA
	update_sprite_all	
	clear_SP	Reset sprite data in PPU

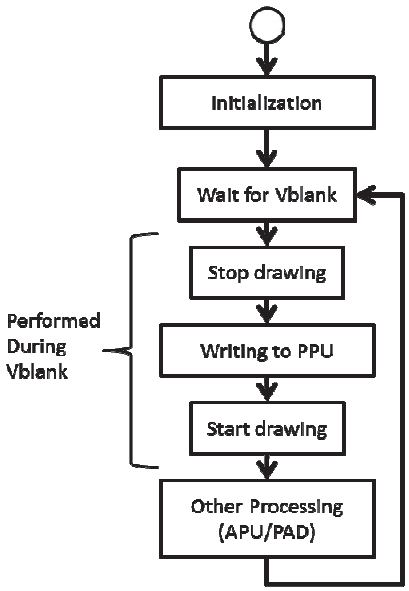

Fig.3 Basic flow of NES program

```
void setBG(unsigned char x,unsigned char y,unsigned
char chr_no){
        int ad;
        ad=0x2000+((x&0x20)<<5)+
            (x&0x1F)+((y&0x1F)<<5);
        *(PPU+6)=(char)(ad>>8);
        *(PPU+6)=(char)(ad&0xFF);
        *(PPU+7)=chr_no;
}
```

Fig.4 Method of setBG to write image(chr_no) to (x,y)

```
void update_sprite(unsigned char num){
        unsigned char tmp;
        tmp=num<<2;
        *(PPU+3)=tmp;
        *(PPU+4)=SP_DATA[0+tmp];
        *(PPU+4)=SP_DATA[1+tmp];
        *(PPU+4)=SP_DATA[2+tmp];
        *(PPU+4)=SP_DATA[3+tmp];
}
void update_sprite_all(){
        *(DMA)=SP_ADDR;
}
```

Fig.5 Method of update_sprite
to write SP_DATA to PPU

BG の操作は特にアドレスの指定や書き込み手順が多い．setBG メソッドを Fig.4 に示す．書き込みに必要なアドレス計算やメモリ書き込みをまとめることで，BG データを書き込むという処理そのものを操作として行った．スプライトは一画面 64 個表示することができる．PPU ライブラリではスプライト描画に必要なデータを格納する SP_DATA 配列を用意した．SP_DATA 配列はファミコンのスプライトデータの仕様に沿った格納を行うため，アドレス指定して行う通常書き込みだけでなくデータ一括転送である DMA を用いて書き込むことができる．我々は通常書き込みと DMA 書き込みそれぞれの処理をまとめ，メソッドを作成した．作成したメソッドを Fig.5 に示す．update_sprite では指定した番号のスプライトを SP_DATA から PPU へ書き込む．update_sprite_all では SP_DATA 内のすべてのスプライトを DMA で PPU に書き込む．このようにポインタを用いずに PPU や画面描画のルールに沿ったプログラミングが可能になっている．

(2)APU

ファミコンは方形波，のこぎり波，三角波，ノイズの 4 つの音を生成することができる．APU ライブラリでは，方形波とノイズの 2 音を用意し，APU への操作を行う．方形波の生成メソッド ch_sq1 を Fig.6 に示す．音の生成は音の長さ，周波数，音量，デューティー比やスイープなどの細かいパラメータ設定が必要である．APU へ書き込むデータはビットごとにパラメータが分かれている．APU ライブラリでは音の生成をわかりやすくするために，引数としてパラメータを渡すことで音を生成する処理単位でメソッドを作成した．本来 APU は音の長さをテーブルとして持っており，パラメータはインデックスで渡すため，時間で指定ができない．作成したメソッドではインデックスから時間を計算し，音の生成時のパラメータは時間指定を可能にした．インデックスで計算できない音の長さは学習者が vblank に合わせて 1/60 秒をプログラムで計測することで用意したパターン以外の長さも可能となっている．

```
void ch_sq1(unsigned char length,unsigned char
freq,unsigned char volume,unsigned char duty,char
sweep){
    *(APU_CHANNEL)|=0x01;
    if(length==0)
        *(APU_SQU1)=(duty<<6)|(volume&0x0f)|
                    0x30;
    else *(APU_SQU1)=(duty<<6)|(volume&0x0f)|
                    0x10;

    ……

    *(APU_SQU1+2)=freq;
    *(APU_SQU1+3)=calc_length(length);
}
```

Fig.6 Method of ch_sq1 to generate sound

(3)PAD

PAD の入力は読み取り開始信号を PAD へ書き込んでからボタンデータをポインタを用いて順番に読み込んでいく．PAD ライブラリではポインタを用いずにボタン情報取得という処理でまとめた．読み込んだデータを keys 配列に格納し，配列でボタンデータを参照するようにした．データ読み取りを行う fetch_keys メソッドを Fig.7 に示す．我々はこのメソッドの機能をデータの読み取りのみのサポートとした．読み取るタイミングや読み取る回数によってチャタリングなどのデータ不良が起きる可能性がある．これらの問題は学習者がプログラム中で解決しなければならない．これにより，組込みシステムの入力装置の制御を学ぶことができる．APU ライブラリと PAD ライブラリとして作成したメソッドを Table5 に示す．

```
void fetch_keys(){
        char i;
        *PAD1=1;
        *PAD1=0;
        for(i=0;i<8;i++){
                keys[i]=*PAD1&0x01;
        }
        return;
}
```

Fig.7 Method of fetch_keys to read button data to keys

Table5 Method list of APU and PAD library

Library	Method	Detail
APU	ch_sq1	Play square wave channel
	stop_sq1	Stop square wave channel
	ch_noise	Play noise channel
	stop_noise	Stop noise channel
PAD	fetch_keys	Read from PAD data into keys

Table6 Sample program list

Sample program	Detail
bg1.c	To display the 'A' to all BG
bg2.c	bg1.c + scroll
bg3.c	To update BG data
sprite1.c	To display the sprite #0
sprite2.c	To display four sprite
sprite3.c	To auto move the sprite #0
pad.c	To move sprite by button
sound1.c	Play square wave channel
sound2.c	Play noise channel
shooting_game1.c	When the button is pressed to display the sprite #0 that move automatically
shooting_game2.c	To display the enemy(BG) and play music
shooting_game.c	shooting_game1.c + shooting_game2.c + Reset the BG data of the sprite place

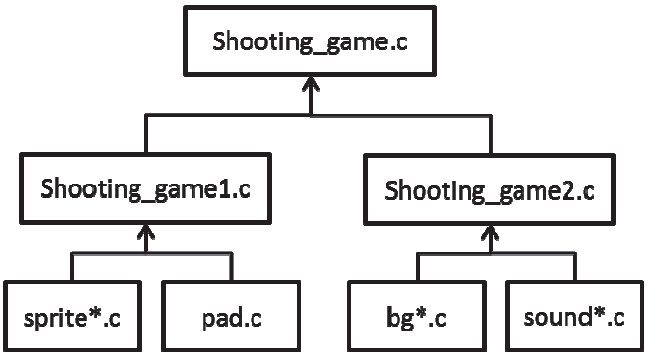

Fig.8 Sample program
The top module includes bottom modules

```
init();
while(1){
        waitvblank();

        prev_a=keys[BT_A];
        fetch_keys();
        if(prev_a==0&&keys[BT_A]==1){
                ch_sq1(2,0x9f,15,2,0);
        }
        if(keys[BT_B]==1){
                ch_sq1(2,0x9f,15,2,0);
        }
}
```

Fig.9 source code of sound1.c

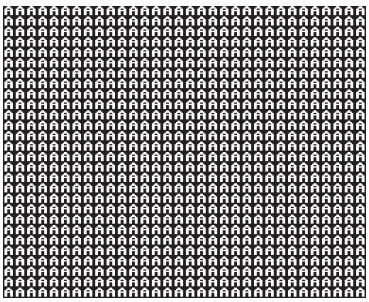

Fig.11 output of bg1.c

```
init()
while(1){
    waitvblank();
    ppu_off();
    if(display==1){
        for(x=0;x<32;x++){
            for(y=0;y<30;y++){
                setBG(x,y,0);
            }
        }
        display=0;
    }
    else if(display==2){
        for(x=0;x<32;x++){
            for(y=0;y<30;y++){
                setBG(x,y,'A');
            }
        }
        display=0;
    }
    set_scroll();
    ppu_on();

    fetch_keys();
    if(keys[BT_A]==1){
        display=1;
    }
    if(keys[BT_B]==1){
        display=2;
    }
}
```

Writing to PPU

Other process

Fig.10 source code of bg1.c
bg1.c fill the screen with 'A'

2.2 サンプルゲーム

サンプルゲームとしてシューティングゲームを選択した．シューティングゲームは PPU の要素である BG や SP と PAD からの入力，APU によるサウンド生成を用いるため，開発したライブラリをすべて使い学習を行うことができる．サンプルはライブラリの機能ごとに用意し，サンプルを組み合わせるとシューティングゲームが完成するように設計した．サンプルの概要を Table6 に組み合わせを Fig.8 に示す．
シューティングゲームの要素は自機や弾を表現するためのスプライト，敵を描画するための BG，発射動作や自機の移動のための PAD 制御，効果音出力のためのサウンドとなる．サンプルはそれぞれの要素に必要なライブラリの使用例となっている．Fig.9 に sound1.c，Fig.10 に bg1.c の一部を示す．sound1.c では A ボタンまたは B ボタンが押された時"ファ"の音を 0.5 秒鳴らすプログラムとなっている．ch_sq1 メソッドの引数を変更することで APU への書き込むデータから音が変化することを確認できる．また，B ボタンは押下されている間，音を生成し続けるため，音が 0.5 秒鳴り終わる前に次の音がなってしまう．A

Fig.12 Run of development game on the original NES

ボタンでは立ち上がり検知処理を用いることで押下時に一回0.5秒音を鳴らすプログラムとなっている.

bg1.c では A ボタンで BG の画像データをすべて0(描画無し)に変更し，B ボタンですべてのタイルに'A'を描画する．PPU の仕様にそって画面書き込み処理と PAD データ取得などのその他の処理を分けて書かれていることがわかる．Fig.11 に bg1.c の実行結果を示す．このようにサンプルプログラムではそれぞれのライブラリの使い方の他にハードウェア上の制約や意図せず起こってしまう処理などの実例を上げて解説している．学習者はサンプルプログラムやプログラム解説，ハードウェア解説を参考にハードウェアの動作を学習することができる.

完成したゲームはファミコン実機で動作する形式で生成される.開発したゲームが実機で操作している様子を Fig.12 に示す.

3. 実施

教材の学習効果の検証のため，開発した教材を用いた授業を企画した．工業高校生 17 名を対象として授業内容の調整を行った．授業当日までのスケジュールを Table7 に示す．高校側からの要望として学年を問わず募集のためプログラミングスキルに差があること，授業は一日で完結するようにすることの 2 点があった．プログラミングスキルは 1 年生 4 人がプログラミング初学者で条件判断文と繰り返し文の学習中，2 年生 11 人がポインタの学習を行っておらず配列の学習済，3 年生 2 人がポインタやプリプロセッサ，関数など通常の参考書の構文は習得済である．当日の授業スケジュールを Table8 に示す．授業時間が短く個人ではゲームの完成までの時間が足りないと考え，2〜3 人のグループに分かれてゲーム開発を行い，グループ開発の中で各自サンプルの動作を確認し合いながらの学習形式をとる．サンプルの解説やファミコンに関する資料を増やし，ソースコードの値変更のみの作業から，ライブラリの詳細説明までを重点的に強化した.プログラミングスキルごとに教材を利用できるように Table3 にあわせて学習者のグループわけを行った．今回は 3 人グループが 5 つ，2 人グループが 1 つの 6 グループに分けた.

Table7 schedule of pre-award meeting

Date	Detail
2015/3/20	Planning
2015/5/8	Demand Research
2015/8/6	Final adjustment

Table8 schedule of lesson

Period	Time	Detail
First	11:05~12:35	Lecture
Second	13:25~14:55	Training
Third	15:10~16:40	

授業当日は大学院生 2 名が各グループの教育補助(TA)についた．1 時間目の全体講義では前半は組込みシステムの概要について，後半はハードウェアについての解説とソースコードレビューから組込みシステムやファミコンに関する基礎知識の学習を行った．2 時間目から 3 時間目は 1 時間目で学習した内容を踏まえてサンプルの動作からグループごとにゲームを開発する．ゲーム開発ではほぼすべてのグループが初めにサンプルの動作を各自で行い，ライブラリやサンプルを使ってできることの把握を行っていた．その後，グループ内で作りたいゲームの方向性を定めて，完成に必要な要素の調査を行い，開発へと進んだ.

学習者の作成したゲームのソースコードを Fig.13 に，実行結果を Fig.14 に示す．このグループはポインタ等の高度なスキルを学習していない初学者である．(1)部分にてシューティングゲームの弾の発射音を複数音組み合わせ，よりリアルな音を作成した．また，(2)にてゲーム終了後に文字を表示するように追加している．サンプルやライブラリ資料から拡張したい機能の仕様を調べ，追加することができている．グループメンバーの話し合いにて，終了時メッセージを表示する際に画面がちらつく現象が起きることが話題になっていた．メッセージの表示処理が多く PPU の書き込み時間が超えているためである．修正までいたらなかったが，グループ内の話し合いで「文字の表示を同時に行いすぎなのではないか？」など，原因に対して考察をすることができていた.

```
while(1){
        waitvblank();
        update_sprite_all();
        if(SP_DATA[0]<50){
                ppu_off();
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                gazou=getBG(del_x,del_y);
                if(gazou==0x4A){
                        setBG(del_x,del_y,0);
                        ch_sq1(3,0x70,13,3,-1);
                        SP_DATA[0]=0;
                        SP_DATA[3]=0;
                        count++;
                }
                set_scroll();
                ppu_on();
        }
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        if(count==16&&SP_DATA[20]<50){
                setBG(4,15,'T');
                setBG(5,15,'H');
                setBG(6,15,'A');
                setBG(7,15,'N');
                setBG(8,15,'K');
                setBG(10,15,'Y');
                setBG(11,15,'O');
                setBG(12,15,'U');
                setBG(14,15,'F');
      (2)       setBG(15,15,'O');
                setBG(16,15,'R');
                setBG(18,15,'P');
                setBG(19,15,'L');
                setBG(20,15,'A');
                setBG(21,15,'Y');
                setBG(22,15,'I');
                setBG(23,15,'N');
                setBG(24,15,'G');
                setBG(25,15,'!');
                setBG(26,15,'!');
        }
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        if(prev_a==0&&keys[BT_A]==1){
                ch_sq1(4,0x7e,15,4,3);
      (1)       ch_noise(2,10,5);
                ch_noise(4,12,11);
        }
}
```

Fig.13 game1 source code
“Shooting game”

```
void msg(char x, char y, char *text){
        while (*text != '¥0'){
                setBG(x, y, *text);          (2)
                x++;
                text++;
        }
}
int main(){
        init();
        waitvblank();
        ppu_off();
        msg(0, 1, "Sequencer");
        msg(0, 2, "v('w')v");
        msg(0, 7, "CH    0    1");
        ppu_on();
        Play();
        while (1){
                waitvblank();
                ppu_off();
                //----Start Drawing
                setBG(4, 8, 0x80 * (ch1[ptr] != RR));
                setBG(7, 8, 0x80 * (ch2[ptr] != RR));
                //----Stop Drawing
                set_scroll();
                ppu_on();
                if (isPlaying) onSequence();
        }
        return 0;
}

void onSequence(){
        if (timer > wait){
                PlayTone();
                IncPtr();
        }
        timer++;
}

void PlayTone(){
        //Ch1
        if (ch1[ptr] == RR) stop_sq1();
        else ch_sq1(0,ch1[ptr],volch1,dt1,0);
        //Ch2
        if (ch2[ptr] == RR) stop_sq2();              (1)
        else ch_sq2(0,ch2[ptr],volch2,dt2,0);
}
```

Fig.15 game2 source code
“Sound sequencer”

別のグループのゲームのソースコードを Fig. 15 に，実行結果を Fig. 16 に示す．このグループはポインタや関数などのスキルを習得しており，ゲーム開発開始段階で，サウンドについて興味が強く，APU のライブラリを深く調査を行っていた．授業中盤に目標とする音を出すためには用意されたライブラリのチャンネル数では不足していることに気づき，APU ライブラリを改変してサウンド出力を増やした (1)．映像出力に関してもライブラリを操作するための関数を追加している．PPU ライブラリでは，BG のタイルが一文字ずつしか変えることができないが，文字列を指定した座標から順番に自動で配置するプログラム

Fig.14 output of game1

を作成している(2).

その他のグループも，用意した画像セットの変更や音楽の作成のための分周期作成などハードウェアの制約を考えながらゲームの作成ができていた.

4. 考察

授業アンケート結果をFig.17に示す．授業を受けて楽しいと思った受講者が 58%，勉強する気持ちが増えたと思った受講者が 70%となった．すべての受講者が授業が楽しかったと答え，本授業が学習者の学習意欲向上につながっていると言える．更に，ハードウェアについて理解できたと答えた受講者が70%，ハードウェアの操作を実感できた受講者が 82%となった．詳しい理由として，「ハードウェアの性能を考えなければ行けなかった．」，「サンプルから引用してプログラムを拡張することができた．」という意見があり，教材を用いてハードウェアを意識した教育が行えている．ハードウェアの操作が実感できなかった意見として，「実際に実機で動かしてないから．」という意見が多く見られた．授業の難易度が難しいと答えた受講者が半数を超えた．理由として，「処理性能の制限でゲームがなかなか動かなかった．」など，教材の使いにくさではなくハードウェアの制約に関する難易度について述べている．このことからもハードウェアの動作や制限を受講者に意識させることができていると言える．TA が受けた質問の多くはファミコンのハードウェア仕様に関するものだった．反して，ライブラリやサンプルは各自仕様を読み取れていたため，質問は少なかった．また，受講者の感想もハードウェアの難しさや質問して初めて理解できたことがあげられていた.このことから，ライブラリやサンプルの資料は十分な量を用意できていたが，ファミコンのハードウェアに関する資料がやや不足していると思われる.

完成したプログラムはサンプルやライブラリの組合せやライブラリの改造など各グループのスキルレベルごとにすべてのグループがハードウェアを操作し，ゲームを完成させていた．このことから，本教材はスキルレベルに応じてハードウェアの学習を行えていたと言える.

Fig.16 output of game2

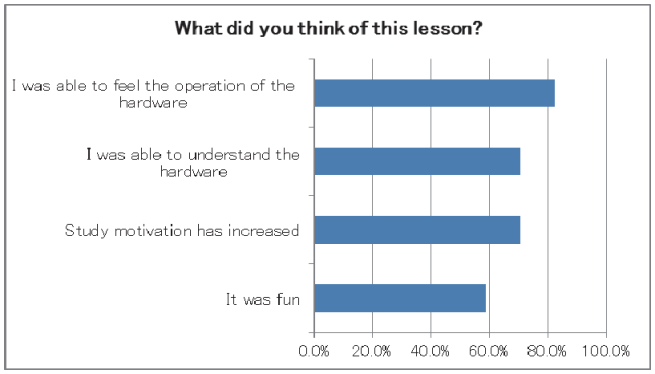

Fig.17 Result of questionnaire

5. おわりに

本研究では実在するゲーム機を題材としたモチベーションを向上させながら組込みシステムの導入教育を行うことのできる教材の開発とその教材を用いた授業を企画した．高校生 17 名に対し授業を行い授業に関するアンケートから教材の教育効果を考察した．結果として，82%の受講者がハードウェアの操作を実感し，70%の受講者が勉強する意欲を増すことができた．さらにすべての受講者が楽しかったと答えた．このことから，開発した教材を用いた授業が組込みシステム教育においてモチベーションの維持とハードウェアの理解へ効果的であることが示された.改善点として，企画した授業スケジュールでは学習者の作成したゲームをファミコン実機で動作させていなかったが，授業最後に ROM へデータを移し，実機での動作を行うことでハードウェアの実感を高めると共に，エミュレータを用いてデバッグしテスト完了後に実機で動作させるという開発プロセスの学習へとつなげることができる.

今回はプログラミング学習経験者が多かったが，プログラミング未経験者や組込みシステム学習初学者など様々な対象への効果の調査を目標とする．今後の課題として，ライブラリを含めた教材や授業企画を第 3 者が利用し，組込みシステム教育を行うことのできるようにパッケージ化を今後の課題とする.

参考文献

1) 長島 和平，長 慎也，「Tonyu System2 ゲーム制作を通じたプログラミング学習に適したフレームワーク」，

情報処理学会研究報告 コンピュータと教育研究会報告 2015-CE-129(2), 1-8, 2015
2) 中村孝, 「ゲームプログラミングを題材としたプログラミング教育の試み」, 情報教育シンポジウム2001論文集 2001(9), 23-26, 2001
3) 栗山 裕 , 橋下 友茂 , 山下 利之, 「ゲームプログラミングによる情報教育の評価方法」, 日本教育工学会論文誌 29(suppl), 181-184, 2005
4) 小林 健一郎, 「プログラミング教育におけるゲームプログラム」, 静岡産業大学情報学部研究紀要 15, 1-16, 2013
5) 独立行政法人 情報処理推進機構, ″組込みスキル標準ETSS2008″, http://www.ipa.go.jp/files/000023848.pdf, 2008, アクセス2015-09-20
6) 独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター, 「新板 組込みスキル標準ETSS 概説書」, 2009
7) 上村雅之, 「ファミコンメディアその技術背景について」, 『計測と制御』第29巻第6号, 551-556, 1990
8) cc65 -the 6502 C compiler, http://www.cc65.org/, アクセス2015-09-2