한국산학기술학회논문지
Vol. 5, No. 4, pp. 292-297, 2004

GVM기반의 모바일게임 프로그래밍

백현옥 · 김재훈 · 남윤성 · 이재욱 · 조태경

A Study on Mobile Game Programming based on GVM

Hyun-Ok Pack, Jae-Hun Kim, Yun-Sung Nam, Jae-Wook Lee and Tae-Kyung Cho

요 약 본 논문에서는 애플리케이션이 하드웨어에 영향을 받지 않고 실행될 수 있게 한 환경을 의미하는 VM(Virtual Machine)과 SKT에서 채택한 플랫폼인 GVM(General Virtual Machine)의 기술내용을 조명하고, GVM을 기반으로한 포트리스 모바일 게임을 개발하였다. 게임은 모바일 C로 구현하였으며, 모바일 C의 특징인 Event Driven방식 게임의 실행과정을 flowchart로 나타냈다.

Abstract This paper is demonstrated about VM that means a circumstance where a application is no dependence of hardware and GVM which is selected by SKT as flatform, thus we developted portris-mobile game based on GVM. this game was embodied(actualized or given to actualized form) by Mobile C and process procedure of Event Drivnen type game that is a characteristic of Mobile C is showed by flowchart.

Key Words : VM, GVM, Mobile Game Programming

1. 개 요

현재 우리가 제공받을 수 있는 모바일 서비스는 몇 개라고 딱 집어 말할 수 없을 만큼 다양하다. 예년엔 없던 기술이 올해에는 개발이 되고 우리가 알고 있는 서비스, 미처 알지 못하는 서비스의 종류도 나날이 늘어나고 있다. 또한 그것은 우리생활에 점점 밀접하게 연관되어 다가오고 있다. 이처럼 모바일이 주목 받고 있는 것은 사용자들에게 접근할 수 있는 분야가 다양하기 때문이다. 예전부터 꾸준히 인기를 얻어오고 있는 서비스로는 벨소리와 그림, 그리고 게임을 예로 들 수 있다[1].

본 논문에서는 모바일 컨텐츠를 개발하기 위한 대표적인 개발 환경 중 하나로 국내 업체에서 개발되어 현재 가장 많이 사용되고 있는 GVM을 기반으로 게임을 개발하였으며, 온라인 게임으로 쉽게 알려져 있는 포트리스 게임을 구현함으로서, 모바일 C와 게임의 제작과정, 동작 원리에 대해 접근해보았다. 논문의 2장에서는 VM의 원리와 구조에 관해 살펴보고 3장에서는 SKT에서 선정한 VM인 GVM에 대해 알아보았으며, 4장에서는 GVM 기반의 간단한 모바일 게임 프로그램인 포트리스를 예로 제시하겠다. 마지막으로 5장에서는 결론 및 향후 연구수행 해야 할 과제에 대해서 알아보았다.

* 상명대학교 정보통신공학과

2. VM의 원리와 구조

국내의 경우 모바일 게임은 2000년 하반기부터 유료화가 추진되었고 첨단 게임산업협회는 2001년 그 시장 규모가 약 137억원에 이를 것으로 전망했다. 또한 2005년경에는 약 580억원의 시장규모를 이룰 것으로 예상한다[1,5]. 이렇듯 앞으로 국내 모바일 게임은 아시아 지역의 주요시장이 될 것이라는 것은 분명하다고 본다.

2.1 모바일 게임의 유형

모바일 게임의 유형은 그림 1과 같이 다양하며, 네트워크 사용도와 단말기의 기능에 따라 구분해보면 크게 6가지로 나뉜다. Embedded Game은 단말기가 제조될 때부터 게임이 내장되어 있는 형태를 말한다. SMS Game은 문자 메시지를 통해 상호간에 게임을 즐길 수 있는 것이고, Microbrowser Game은 WAP기반으로 간단한 그래픽과 애니메이션을 제공하는 게임을 말한다. Multiplayer Game은 네트워크에 접속해 여러 사람이 서로 대전하여 즐길 수 있는 게임을 말하며, Streamed Game은 네트워크를 통해 실시간으로 즐길 수 있는 게임을 말한다. 마지막으로 Downloadable Game은 단말기에 게임을 다운로드한 후 단말기 단독으로 또는 네트워크에 접속하여 게임을 즐기는 것을 말한다. 가장 일반적인 모바일 게임의 형태라고 할 수 있다[2].

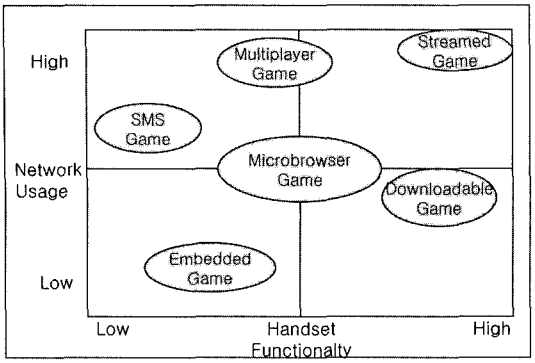

그림 1. 모바일 게임 유형별 포맷

2.2 VM

VM(Virtual Machine)은 애플리케이션이 하드웨어에 영향을 받지 않고 실행될 수 있게 한 환경을 의미한다. VM의 특징은 이기종간의 자유로운 이식성과, 사용자의 추상화 API를 제공한다는 것이다. 그렇기 때문에 동일한 VM이 porting된 단말기에서는 동일한 실행 환경을 가지게 되고, 개발자는 단말기, OS개발 환경과는 무관하게 독립적인 개발환경에서 VM 애플리케이션 개발이 가능하게 되는 것 이다.

VM이 등장하게 된 배경에 대해서 알아보자. 핸드폰과 같은 모바일 device가 가지고 있는 소량의 메모리와 저속의 CPU에서 기인한 원천적인 문제점이 고용량의 DB가 고속으로 처리되어야 하는 모바일 멀티미디어 서비스의 발전을 주춤하게 만들었다. 이러한 제한된 메모리 활용을 극대화하는 방법으로 다운로드와 삭제를 반복하는 구조를 채택해야 했고, 이를 위해 단순 데이터를 넘어 독립적인 실행 구조를 갖는 애플리케이션을 다운로드하고 오프라인 상태에서 이를 반복 실행할 수 있도록 하는 것이 필요했고, 이렇게 해서 등장한 것이 바로 virtual machine, 즉 VM이다.

VM의 등장으로 인해 WAP(Wireless Application Protocol)기반의 브라우징 서비스의 제한된 속도와 비싼 서비스 요금, 안정성 등에 대한 문제를 해결할 수 있었고, 각 이동통신사는 CDMA-2000 서비스를 개시한 시점인 2001년 5월부터 새로운 서비스를 위한 플랫폼으로서 VM을 채택하기 시작했다.

이동통신사별로 채택한 VM은 각각 다르며 그에 따라서 개발 환경도 다르다. 표 1에는 각 이동통신사별 주요 VM을 나타내었다. SK텔레콤은 신지소프트의 GVM과 더불어 XCE에서 개발한 SK-VM을, KTF는 모빌탑의 MAP에 퀄컴의 BREW를, LG텔레콤은 JAVASTATION 서비스에 MIDP를 채택하였다. 이동통신사별로 다른 독자적인 VM을 채택함에 따라 플랫폼 표준화가 쟁점으로 부각되었고 그로인하여 WIPI가 등장하였다[5].

3. GVM (General Virtual Machine)

GVM은 모바일 C를 기반으로 하는 무선 애플리케이션 다운로드 플랫폼으로 SKT에서 채택한 VM을 지칭하는 것으로 단말기 소프트웨어의 UI 태스크에 위치하기 때문에 통화와 같은 단말기의 기본기능에는 영향을 주지 않는다. GVM의 특징으로는 단말기의 리소스 요구를 최소화 한 것과, 단말기에 porting이 쉽고, 모바일 C언어를 사용해 애플리케이션 개발이 용이함은 물론 실행 성능이 우수하다는 것을 말할 수 있다.

그림 2에서 확인할 수 있듯이 GVM은 SDK, Server, GVM모듈이 탑재된 핸드폰으로 구성된다. GVM SDK (Software Development Kit)는 GVM이 탑재된 단말기에서 실행되는 애플리케이션을 작성하는 개발도구를 말

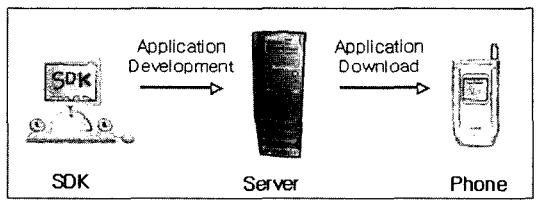

그림 2. GVM시스템의 구성

표 1. 각 이동통신사별 VM

VM 종류	GVM	BREW	JAVASTATION	WIPI
이동통신사	SK텔레콤	KTE	LG텔레콤	3개 이통사
개발업체	신지소프트	퀄컴	SUM(LG)	KWISF
실행환경/사용언어	모바일C 기반, 개발용 SDK 제공	ANSI-C/C++	자바(MIDP)	ANSI-C/C++ 자바 (MIDP)
특징	GNEX출시 3D엔진제공 최대 512kB까지 가능	리코시스와 제휴하여 3D엔진 제공	다른 이통사의 VM에 비해 속도가 느리고 불안정함	2004년 9월쯤 완전 상용화 예정

하며, 모바일 C컴파일러, 미디어툴킷, 에뮬레이터로 구성된다. Server는 애플리케이션 DB를 보유하고 단말기와 접속하여 애플리케이션의 다운로드를 제공하는 기능을 수행한다. GVM모듈은 단말기에 탑재되는 S/W를 말하며, Porting의 용이성이 고려되어 제공이 되며 Event-Driven방식으로 실행하도록 설계된다. 게임 개발자는 SDK를 이용하며 모바일 게임을 제작하고 Server에 등록을 하고 사용자는 Server에 접속해서 다운로드하여 사용하게 된다.

3.1 GVM SDK의 구성요소

SDK는 http://gnexclub.com 에서 다운로드 할 수 있다. 다운로드 할 수 있는 웹페이지엔 *GNEX SDK*와 *GVM SDK*가 있는데 그중에서 본인이 원하는 개발환경의 SDK를 다운로드 하면 된다.

SDK는 크게 모바일 C컴파일러, 미디어툴킷, 에뮬레이터로 구성된다.

모바일 C컴파일러는 모바일 C로 작성된 프로그램을 컴파일하여 오브젝트파일(*.SGS)을 생성한다. SGS파일은 단말기에 다운로드 되어 가상머신에 의해 실행된다. 또한 이것은 에뮬레이터상의 실행파일이 되어 에뮬레이션 할 수 있게 된다. SGS 파일은 내부적으로 헤더영역, 코드영역, 데이터영역으로 구성된다.

컴파일과 빌드를 수행하였을 때 compile error나 warning이 없으면 File Conversion Success!!! 라는 메시지가 뜨며, 그 소스파일에서 사용된 String Data, Image Data등의 개수를 확인할 수 있다. 그림 3에 Mobile C Compiler 실행화면을 나타내었으며 여기서 사용된 Image Data는 7개임을 알 수 있다.

미디어 툴킷은 다시 오디오 툴과 이미지 툴로 나눌 수 있다. GVM SDK에서 제공하는 오디오 툴로는 Buzzer Master와 Audio Master가 있다. Buzzer Master는 음원을 직접 제작하여 모바일 C 프로그램에서 사용할 수 있는 형태로 음원 데이터를 변화 시킬 수 있다. Buzzer Master의 실행화면을 그림 4에다 나타내었다.

그림 3. Mobile C Compiler 실행화면

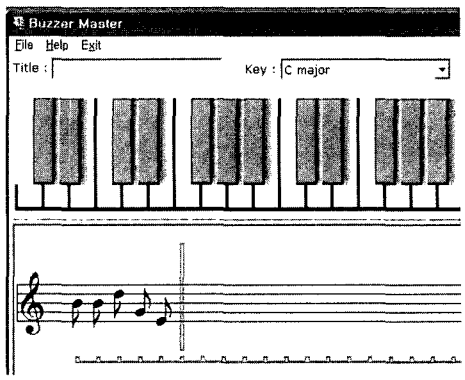

그림 4. Buzzer Master 실행화면

Audio Master는 이미 작성된 음원 데이터를 읽어 모바일 C형태로 변환시켜 준다. 16화음인 MA2음원과 40화음인 MA3음원을 Audio Master를 사용해 converting하면 모바일 C 형태인 *.ssd 파일로 변환되는 것이다. 이렇게 변환된 음원은 소스 프로그램에 include하여 사용하며, 게임 속의 음원은 이러한 과정을 통해서 얻게 되는 것이다.

다음으로 GVM의 이미지 제작과정을 살펴보겠다. 이미지 제작에 사용되는 툴로는 Image Master가 있다. Image Master의 사용방법은 먼저 포토샵, 페인트샵 등의 그래픽 툴을 이용하여 *.bmp, *.gif, *.png 등의 원본이미지파일을 제작한다. 그리고 Image Master에서 원본 이미지 파일을 읽어 들여 모바일 C형태인 *.sbm 파일로 변환하게 되는 것이다. 이것 역시 모바일 C소스 프로그램에 include되어 사용된다.

그림 5는 Image Master를 실행시킨 화면으로 하나의

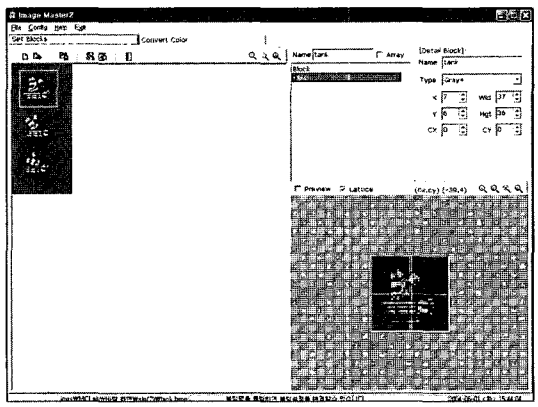

그림 5. Image Master 실행화면

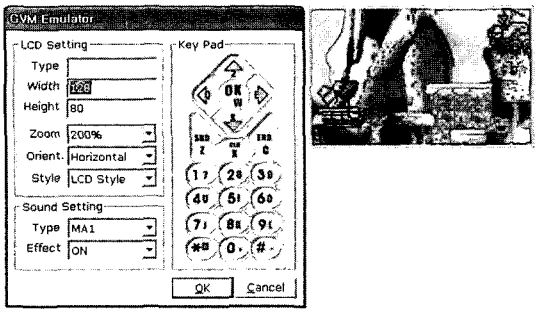

그림 6. 에뮬레이터 실행화면

그림파일을 불러온 것이다. 하나의 그림파일 속에 3개의 탱크 그림이 있는데 그 3개의 그림 중 사용하고자 하는 그림에 마우스로 드래그 하여 선택을 하고 선택한 부분의 그림 이름을 적어준다. 이름을 적지 않았을 경우 배열 형식으로 저장이 되며, 소스프로그램에서 이미지를 사용할 때는 그림의 이름으로 불러서 사용하게 된다. 즉, 하나의 파일을 include했지만 3개의 그림을 각각의 이름으로 사용하는 것이다.

GVM 에뮬레이터는 PC에서 GVM 에플리케이션의 동작을 단말기와 동일한 상황으로 에뮬레이션하고, 단말기의 다양한 하드웨어 사용, RTOS의 이벤트 레벨까지 에뮬레이션 하여 단말기와 동일한 조건으로 동작 시험을 가능하게 한다. 그림 6에 에뮬레이터 실행화면을 나타내었다.

3.2 GVM SDK의 동작

앞서 살펴본 것과 같이 Sound와 Image는 각각 compile되어 모바일 C에서 사용가능한 형태로 변환되며 이것은 프로그램 소스파일에서 include하여 사용한다. 소스프로그램은 Mobile C Compiler를 통하여 GVM의 오브젝트 파일인 *.sgs로 compile되며, 이 파일로 에뮬레이션이 가능해지며, GVM이 탑재된 단말기에서 다운로드 하여 사용하게 되는 것이다. 이것의 흐름을 그림 7에 나타내었다.

3.3 GVM이 제공하는 기능

GVM은 서버에 접속하여 원하는 애플리케이션을 다운로드해 실행할 수 있게 한다. 일단 다운로드한 애플리케이션은 단말기의 메모리에 저장되어 사용자가 삭제하기 전까지는 재 다운로드 없이 수행할 수 있다. 이것은 VM의 등장하게 된 요소이며, 가장 큰 특징이라 할 수 있다. GVM은 모든 종류의 단말기 LCD 및 오디오 장치를 제어할 수 있으며, 다양한 이미지 포맷을 수용한다. 또한 GVM 은 타이머를 이용한 리얼타임 애플리케이션을 실행할 수 있고, 단말기에서 TCP/IP소켓 서

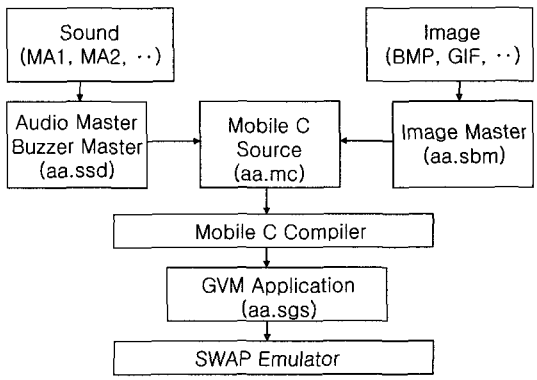

그림 7. GVM SDK의 동작

비스를 제공하는 경우 이를 이용하여 클라이언트-서버 형식의 애플리케이션을 실행할 수 있다. 그리고 GVM에서는 단말기의 Vibrator, KeyTone, BackLight등의 장치를 제어할 수 있는 API를 제공하므로 이를 이용해 생동감 있는 애플리케이션을 실행 할 수 있다. 마지막으로 GVM이 제공하는 기능으로는 시리얼 통신을 가능하게 한다는 것이다. 단말기의 RS-232C 포트를 통해 컴퓨터와 같은 외부 장치와 연결하여 데이터를 주고받을 수 있는 응용기능이 제공된다[7].

3.4 Mobile C의 특징

첫째로 표준 C가 절차적 언어임에 반해 모바일C는 Event-driven방식으로 실행된다. 어플리케이션이 시작, 키패드 입력, Timeout 발생등과 같은 이벤트 핸들러의 집합으로 구성된다. 또한 함수의 재귀순환이 금지되어 있고, data type을 2byte integer로 제한한다. 표준 C에는 없는 image, sound, voc, string과 같은 미디어 type이 추가되었다[2][4].

포트리스 게임에 사용된 이벤트 핸들러로는 EVENT_TIMEOUT, EVENT_KEYPRESS가 있다. EVENT_TIMEOUT은 setTimer에서 지정한 일정시간이 경과했을 때마다 Timeout Event를 발생시켜 해당되는 이벤트를 수행하는 것이다. EVENT _KEYPRESS는 사용자가 단말기의 키패드를 누를 때 발생하는 이벤트이다. 이벤트 발생시 사용자가 누른 키 정보는 swData 값으로 알 수 있는데 1이 입력되었을 때의 swData는 SWAP_KEY_1 또는 0x01이다. 이 값은 SString.h에 정의 되어 있다. SString.h은 GVM SDK에서 제공되는 헤더파일로, LCD를 제어하는 함수, 진동을 제어하는 함수 등 여러 함수들에 관해 정의 되어 있으며, 소스파일을 제작하기 전에 항상 include하여야 한다. swData 값을 이용해서 switch-case 문으로 각각의 키를 나누어서 사용자가 누른 키에 해당되는 이벤트를 수행하게 되는 것이다.

4. 모바일 게임의 구현

모바일 C를 이용해 포트리스 게임을 구현해 보았다. 이 게임은 왼쪽 탱크 캐릭터에서 미사일이 발사되고, 날아가는 미사일의 좌표와 오른쪽 탱크 캐릭터의 좌표가 같아지면 탱크가 터지고 CLEAR!! 라고 출력한다. 또한 맞춘 횟수를 count하여 현재 stage를 출력하도록 구현하였다.

그림 8은 처음 게임을 실행시켰을 때의 에뮬레이터 상의 모습이다. 그림 9는 미사일을 쏘았을 때이고, 그림 10은 미사일이 오른쪽 탱크에 맞았을 때 탱크가 터지는 모습을 capture 한 것이다. CLEAR!! stage: 1이 출력되

그림 8. 초기화면

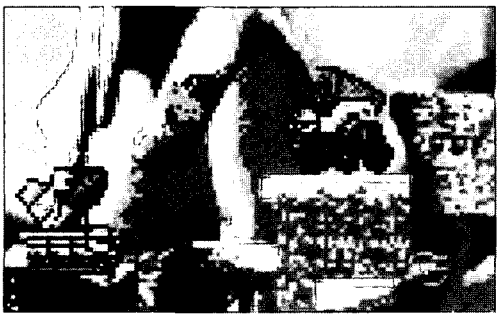

그림 9. 포탄발사

그림 10. 명중

```
void main()
{
      int i = 0, j = 0;
      ClearWhite();
      initFunc();   // 초기화한다
}
```

그림 11. main()함수

었음을 확인할 수 있다

4.1 포트리스 게임의 순서도

포트리스 게임을 크게 3등분하면, main()함수를 비롯한 게임을 구성하고 동작시키는 여러 함수들과, 키패드 입력 시 발생하는 이벤트 처리 부분과, 타임아웃이 발생하였을 때 처리해주는 부분으로 나눌 수 있다.

프로그램을 실행시키면 main() 함수의 initFunc()가 호출된다. initFunc()에는 모든 변수들을 초기화시켜주고 게임이 실행되었을 때의 초기모습을 그려준다. main()함수의 코드를 그림 11에 나타내었다.

다음으로 키패드 입력 시 발생하는 이벤트 처리 부분을 살펴보겠다. 핸드폰의 키패드는 버튼이 입력되었을 때 이벤트가 발생한다. 왼쪽 버튼이 입력되었다면 swData는 SString.h에 정의되어 있는 것과 같이SWAP_KEY_LEFT가 될 것이고 탱크를 뒤로 움직이게 하는 함수 moveBack()과 움직인 탱크를 보여주는 show()함수를 호출하여 사용하게 된다. OK버튼이 입력되었다면 case SWAP_KEY_OK:에 의해 포탄을 발사하는 bombFire()가 호출된다. 키패드 입력 시 발생하는 이벤트를 처리하는 EVENT_

```
void EVENT_KEYPRESS()
{
    if(motion==0)
    {
        switch(swData)
        {
            case SWAP_KEY_LEFT :
                moveBack(); show();
                break;
            case SWAP_KEY_OK:
                if (end == 1) bombFire();
                break;
            case SWAP_KEY_RIGHT :
                moveForward(); show();
                break;
            case SWAP_KEY_RIGHT :
                moveForward(); show();
                break;
            case SWAP_KEY_UP :
                Up(); show();
                break;
            case SWAP_KEY_DOWN :
                Down(); show();
                break;
        }

    }
```

그림 12. EVENT_KEYPRESS()

```
void EVENT_TIMEOUT()
{
//  SetTimer가 작동되었을 경우, 폭탄이미지가 이동시 X, Y좌표를 계산

    st.bombX = x1 + (start * 3) + st.leftX;

    switch(count){
       case 0 : st.bombY = y - (a0[start] * 1);
                break;
       case 1 : st.bombY = y - (a1[start] * 3);
                break;
       case 2 : st.bombY = y - (a2[start] * 3);
                break;
    }

    end = 0;    // 폭탄이 마지막 위치에 도달하지 않음
    start++;    // 폭탄이 이동시 a1배열의 값을 가져오기 위해 1씩 증가

    if (aa ==1 || start >= 40 ) {
       start = 1;          // 초기화
       end = 1;            // 폭탄이미지가 마지막 위치에 있음을 알림
       ResetTimer();       // 타이머를 종료
    }
    show();  // 타이머가 작동될때마다 모든 캐릭터들을 보여줌
}
```

그림 13. EVENT_TIMEOUT()

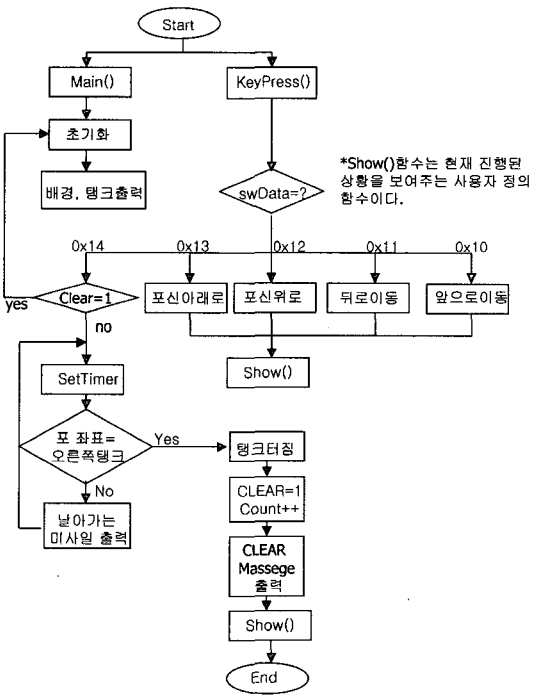

그림 14. 포트리스 게임의 동작 알고리즘

KEYPRESS()의 코드의 일부를 그림 12에 나타냈다.

마지막으로 타임아웃이 발생하였을 때 처리해주는 부분을 보겠다. OK버튼이 입력되었을 때 호출된 bombFire()함수 속에는 SetTimer()를 호출하는 내용밖에 없다. SetTimer()은 millisecond단위로 시간을 지정할 수 있으며 지정된 시간이 지나면 Timeout 이벤트를 발생하게 된다. 실질적으로 포탄을 발사하는 부분은 Timeout이벤트를 처리해주는 EVENT_TIMEOUT()부분으로 이것의 코드를 그림 13에 나타내었다.

키패드의 아래, 위 버튼으로 미사일 각도를 조절하여 오른쪽 캐릭터에 발사하여 맞췄을 경우 그 횟수를 count를 하여 현재 stage를 출력한다. 게임을 다시 시작할 경우 OK 버튼을 누르고 이때 게임이 초기화되어 다시 실행되는 구조로 코딩하였으며, 게임의 전반적인 동작 알고리즘을 그림 14에 도시하였다.

5. 결 론

신지소프트에서는 GVM의 윗 버젼인 GNEX를 출시했다. GNEX는 512 kB로 GVM보다 사용가능한 메모리가 클 뿐만 아니라 3D까지 지원되기 때문에 앞으로 좀 더 눈이 즐거운 게임을 즐길 수 있게 될 것이다.

VM과 SKT에서 채택한 플랫폼인 GVM에 대해 전반적으로 알아보고 모바일 C를 이용해 간단한 프로그램을 구현해 보았다. 표준 C와는 다른, 모바일 C의 특징인 이벤트처리 부분인 KeyPress()나, Timeout()등을 코딩해 보면서 모바일 C와 표준 C의 차이점 그리고 flowchart로 게임이 동작되는 과정을 살펴보았다. 향후에는 게임 뿐 아니라 영어 학습 프로그램, 원격제어프로그램 등에 관한 연구를 수행할 예정이다.

참고문헌

[1] 유소란, "모바일 게임 시장 및 개발 동향", 한국 정보처리학회, 제9권 3호, pp. 42-48, 2002. 5.

[2] 김정훈, "Mobile Game Contents Project", 베스트 북, 2002.

[3] 이정환, 수주호, "GVM&MAP 모바일 프로그래밍", 대림, 2002.

[4] 앤슬래시닷컴 저, "GVM Programing", 삼양출판사, 2001.

[5] 황병연, 오명석, "모바일 게임을 위한 압축기술", 정보처리학회지, 제9권 3호, pp. 49-54, 2002. 5.

[6] 김상택, "Mobile Contents 발전방안", Telecommunictions Review, 제10권 6호, pp.1124-1131, 2000.

[7] 김승훈, "GVM 개발환경에서 모바일 온라인 컨텐츠를 위한 프로토콜", 멀티미디어학회논문지, 제7권 2호, pp. 241-250, 2004. 2.