技術論文

심볼 생성기용 소프트웨어 개발

박덕배*, 이재억*

Development of symbol generator software

Deok-Bae Park* and Jae-Uk Lee*

ABSTRACT

This paper describes the development and implementation for the SYMBOL-GEN(SYMBOL GENERator) software. The SYMBOL-GEN software is for improving graphic processing speed and decreasing data communication load in ASC by generating and downloading off-line symbol file for HUD and MFD, which are the main display equipments in military aircraft. The SYMBOL-GEN is developed on PC using C++ language and MS Visual Studio 6.0 development tool. It is also designed to be modified and extended easily by introducing object-oriented software development technique.

초 록

본 논문에서는 군용 항공기의 주 시현 장비인 HUD와 MFD에 그려지는 여러 가지 심볼 형상들을 off-line에서 효율적으로 생성/편집하여 ASC 그래픽 보드 내부 플래시 메모리의 일정영역에 직접 다운로드 함으로써, ASC의 그래픽 처리 속도 증가 및 데이터 통신 부하를 감소시킨 심볼 생성기용 S/W인 SYMBOL-GEN의 개발내용과 실제 적용 결과를 기술하였다. 개발된 SYMBOL-GEN S/W는 PC상에서 C++ 언어와 MS Visual Studio 6.0 개발환경을 사용하여 구현되었다. 또한, 최근의 객체 지향형 프로그래밍 개발기법을 도입하여 수정과 확장성 등이 용이하도록 설계되었다.

Key Words : SYMBOL-GEN(SYMBOL-GENerator), Head-Up Display(전방상향 시현기), Multi-Function Display(다중기능 시현기), Avionic System Computer(항공전자시스템 컴퓨터), Operational Flight Program(비행운용프로그램), Graphic Process(그래픽 프로세서), Graphic Firmware(그래픽 펌웨어) Symbol Table (심볼 테이블), segment(세그먼트)

1. 서 론

군용 항공기의 주 시현 장비인 전방상향 시현기(Head-Up Display, 이하 HUD)와 다기능 시현기(Multi-Function Display, 이하 MFD)는 조종사에게 각종 항법 정보, 무장상태 정보, 탄도 정

보 및 시스템 관리 정보 등의 다양하고 복잡한 정보들을 숫자, 문자 및 그래픽 형태로 시현 함으로써 조종사의 임무 부하를 감소시켜 효율적인 임무 수행을 가능하게 하는 시현 장비들이다. 그러나 이들 시현 장비들은 수동 영상 모니터로서 동작하고 실질적인 그래픽 영상의 생성 및 연동은 중앙통제형 컴퓨터인 항공전자시스템 컴퓨터(Avionic System Computer, 이하 ASC)에서 이루어진다. ASC는 HUD 및 MFD용 그래픽 영상 생성과 연동기능 이외에 항공기에 탑재되는 INS/GPS(Inertial Navigation System/Global

† 2003년 2월 13일 접수 ~ 2003년 8월 30일 심사완료

* 정회원, 국방과학연구소

연락처, E-mail : deokbaepark@yahoo.co.kr

대전시 유성우체국 사서함 35-3호

Position System), ADC(Air Data Computer), TACAN(Tactical Air Navigation)등과 같은 항법 센서와 연동기능, ACP(Avionics system Control Panel), WCP(Weapon Control Panel) 등과 같은 조종사 명령 입력처리기능, 탄도계산 및 항법 계산 등의 임무 데이터 처리기능을 수행한다[1]. ASC는 통합기능을 구현하기 위하여 각 기능별로 모듈화 된 하드웨어 보드들과 그 제어를 위한 라이브러리 프로그램인 SCIL(Standard Computer Interface Library), 그래픽펌웨어(Graphic Firmware) 및 응용 S/W인 OFP(Operational Flight Program)등으로 구성된다.

ASC의 HUD 및 MFD용 그래픽 영상신호 기능 중에는 off-line에서 심볼 생성기용 소프트웨어를 사용하여 숫자, 문자, 도형 등을 미리 지정된 심볼 명칭과 특징들로 코드화 된 심볼 데이터 구조체 파일을 생성하고 라스터(Raster) 그래픽 영상 생성 모듈인 그래픽 프로세서(Graphic Processor, 이하 GP) 보드 # 1과 커시브(Cursive) 그래픽 영상 생성 모듈인 GP 보드 #2 각각의 플래시 메모리에 다운로드 함으로써, 응용 S/W인 OFP에서 직접 심볼 그리기를 수행하지 않고, 위치, 크기, 회전, 속성 등의 요소와 함께 Symbol Table 참조 포인터를 전달하여 데이터 통신부하 및 그래픽 처리속도 향상을 도모하는 기능이 있다.

본 논문에서는 심볼 생성기용 S/W인 SYMBOL-GEN의 설계 내용 및 적용 결과에 관하여 소개하고자 한다. SYMBOL-GEN는 심볼 세트의 개발 및 수정을 위한 소프트웨어로, 완성된 심볼 세트는 두 개의 파일로 출력된다. 심볼 명칭을 선언한 ada용 데이터 package인 sym_def_ada 파일은 ASC의 주 운용 프로그램인 OFP를 개발할 때 하나의 패키지로 포함된다. 또한 이진 형태의 *.xxx 파일은 SYMBOL-GEN의 다운로드 기능에 의해 각 GP 보드 플래시 메모리의 Symbol Table 영역에 내장(embedded)되어 진다. 개발된 SYMBOL-GEN S/W는 PC상에서 구현되었으며 최근의 객체 지향형 프로그래밍 기법을 도입하였다. 또한, S/W 개발 언어는 C++ , 개발환경은 MS Visual Studio 6.0이 사용되었다.

II. SYMBOL-GEN 설계

2.1 심볼 생성을 위한 하드웨어 운용 환경

ASC는 모듈화 된 다목적 컴퓨터로서 총 10개의 6U 보드를 탑재하고 있으며, 이들의 구성 및

그림 1. ASC 내부 전자회로 구성

그림 2. GP보드 내부구성 및 데이터 흐름도

인터페이스는 그림 1과 같다. 그림과 같이 HUD 및 MFD용 그래픽 영상생성을 위하여 ASC 내부에는 각각 그래픽 프로세서(Graphic Process, 이하 GP) 보드 # 1 모듈과 GP보드 # 2 모듈이 있다. GP보드 # 1 모듈과 GP보드 # 2 모듈은 하드웨어적인 형상은 동일하나 그래픽 펌웨어가 CGF(Cursive Graphic Firmware)와 RGF(Raster Graphic Firmware)로 구분되어 설계되었다.

GP보드의 하드웨어 구성은 그림 2와 같다. GP 보드는 32비트(0xFFFF FFFF)의 주소 영역을 가지며, 이는 메모리 및 I/O 리소스들에 그림 3과 같이 할당되어 있다. EPROM 영역에는 모니터 프로그램이 내장되어 시스템 부팅, 모니터 및 그래픽 펌웨어를 플래시 메모리에 다운로드 하는 기능을 수행하고, 플래시 메모리에 다운로드 된 그래픽 펌웨어는 부팅 완료 시, 그래픽을 위한 각종 프리미티브(primitive) 및 그래픽 시스템 제어를 수행한다. 플래시 메모리의 일부인 Symbol Table 영역에는 심볼 생성기용 소프트웨어인 SYMBOL-GEN에 의해 생성/다운로드 된 심볼 데이터 구조체 코드가 있다. Arc Table 영역은 원호 그리기에 수행되는 반복적인 실수 및 곱셈 연산의 부하를 줄이기 위해, 초기화 시 미리

\$FFFF FFFF	Processor Reserved
\$FFD0 0000	Reserved
\$C000 0000	Reserved
\$4000 0000	Graphic Area
\$3000 0400	Peripheral Control
\$3000 0000	Local DPRAM
\$2080 0000	Reserved
\$2008 0000	Spare SRAM
\$2003 0000	Arc Table
\$2000 0000	Local SRAM
\$1008 0000	Not Used
\$1007 0000	Symbol Table
\$1000 0000	Flash(Firmware)
\$0000 2000	Reserved
\$0000 0000	EPROM(Monitor)

그림 3. GP보드의 메모리, I/O 맵

Symbol Table의 원호 성분을 검색하여 SRAM의 특정영역에 테이블 형태의 데이터 스트럭처를 구축하기 위한 영역이다. SRAM영역은 데이터 영역으로 그래픽 펌웨어가 구동 시 변수 및 상수가 생성 또는 저장되는 영역이다. Peripheral Control 영역은 I/O 주소 영역으로서 그래픽 시스템의 통제를 위한 각종 레지스터 및 보조 기억장치로 사용된다. Graphic Area는 라스터 및 커시브 생성보드의 VRAM 및 SRAM 영역이며, DPRAM은 디스플레이 파일 버퍼 영역이다[2].

2.2 심볼 생성을 위한 소프트웨어 운용 환경

CPU보드의 OFP/SCIL 소프트웨어가 생성한 디스플레이 파일(Display File)은 데이터 패키지를 전달하기 위한 데이터 스트럭처로써, GP보드의 인터페이스 버퍼인 DPRAM(Dual Port RAM)에 20msec마다 전달된다[2]. GP보드의 그래픽 펌웨어는 DPRAM에서 디스플레이 파일을 한 워드씩 읽고 해독하여 심볼 그리기 명령인 경우에는 SYMBOL-GEN에 의해 미리 코드화 된 심볼들이 플래시 메모리의 Symbol Table로부터 참조되어 그려진다. 심볼이 그려지는 세부적인 과정은 다음과 같다.

1) CPU상에서 OFP 실행중 DRAW_SYMBOL(SYM_DEF.심볼명) 코드를 만나면 SCIL 내 WFG_IF_SYMBOL 패키지의 procedure DRAW_SYMBOL 함수가 호출되어 실행되어 그림 4와 같은 디스플레이 파일 구조에 DRAW_SYMBOL(심볼번호) 부분이 생성되어 DPRAM에 전달된다.

2) GP보드의 그래픽 펌웨어는 디스플레이 파일

그림 4. 디스플레이 파일 구조

그림 5. Symbol Table 구조

중에 DRAW_SYMBOL 부분을 만나면 그래픽 펌웨어의 process_draw_symbol 함수를 실행시킨다.

3) process_draw_symbol 함수는 다시 process_macro_call 함수를 호출/실행하여 심볼명(Symbol Table의 순서)을 뽑고 이로 부터 그림 5와 같은 Symbol Table의 Look-UP 영역에서 심볼의 시작번호를 추출한다.

6) 이후, 이 추출된 심볼의 시작번호로 가서 Symbol Table의 Data 영역에 저장된 해당 심볼의 코드를 추출한 후, 심볼의 각 segment별로 해당함수를 실행시켜, 미리 저장된 그리기 정보로부터 심볼들이 빠르게 그려지도록 한다.

2.3 SYMBOL-GEN 개요 및 구성

SYMBOL-GEN은 off-line에서 심볼을 형상화하고 형상화된 심볼을 코드화하며, 이 코드화 된 심볼 파일을 GP보드 플래시 메모리 내 Symbol Table 영역에 내장하기 위해서 사용된다.

그림 6. SYMBOL-GEN 구성도

그림 7. SYMBOL-GEN 초기화면

SYMBOL-GEN은 객체화된 클래스들로 구성되어 있으며 각 클래스간의 연결은 그림 6과 같다. SYMBOL-GEN은 기존의 심볼 파일을 입력받거나 새로운 파일을 생성할 수 있으며, 심볼을 그래픽 모드 또는 텍스트 모드로 생성/편집 할 수 있다. 생성된 심볼들은 플래시 메모리에 다운로드 가능한 이진 형태의 파일과 Ada용 데이터 패키지인 sym_def_ada 파일로 저장될 수 있다. 생성된 이진 코드화 파일은 ASC GP보드 플래시 메모리의 Symbol Table 영역에 RS 232 통신포트를 통해 직접 다운로드 할 수 있다.

2.4 SYMBOL-GEN 상세 기능 및 알고리즘

초기화면

SYMBOL-GEN 실행파일을 작동시키면 아래 그림 7과 같은 초기 화면이 보여진다.

파일 열기/저장

심볼을 구성하는 기본단위는 segment(세그먼트)로써, CSegment 클래스에 의하여 객체화 된다. CSegment 클래스는 세그먼트 타입 (INVISIBLE, SOLID, CW, CCW, MACRO)을 데이터 값으로 가진 열거형 변수 m_SegmentType과 세그먼트의 시작점과 끝점의 x, y 좌표 값을 각각 저장하는 CPoint형 m_pointFP, m_pointSP 및 CString형 m_strMacro 변수를 데이터멤버로 가진다. Segment로 구성된 심볼은 다시 CSymbol 클래스에 의해 객체화되며, CSymbol 클래스는 심볼명을 저장하는 m_strSymbol과 심볼별로 심볼을 구성하는 Segment의 총 개수를

그림 8. 심볼용 클래스 상호관계

저장하는 `m_nNoSegment` 및 `CArray`형 `m_segmentArray`를 데이터 멤버로 가지는데, 특히 `m_segmentArray` 변수는 심볼을 구성하는 각 `segment`의 특성 즉, 세그먼트 타입, `first point` (시작점), `second point`(끝점) 좌표 값들을 저장한다. 또한 `CSymbol` 클래스로 객체화된 각 심볼의 포인터들은 `CTypedPtrList` 템플릿을 사용하여 `CSymgenDoc` 클래스의 `m_symbolList` 데이터 멤버에 저장함으로써, `CSymgenDoc` 클래스가 `CSymbol` 클래스를 참조하도록 하였다. 아래 그림 8은 `Csegment`, `CSymbol`, `CSymgenDoc` 클래스의 데이터 멤버간 상호관계를 나타낸다.

`CSymgenDoc` 클래스의 `m_symbolList` 변수에 저장된 심볼 정보들은 이진 포맷 형태인 `*.sym` 파일 또는 `text` 포맷 형태인 `*.sxt` 파일로 저장 가능하다. 이에 `SYMBOL-GEN`에서는 파일 열기/저장을 위한 두 가지 방식이 가능하다. 첫째는 파일메뉴의 `OPEN` 또는 `SAVE` 메뉴를 사용하여 자동으로 `*.sym` 파일을 열기 및 저장하는 방식이며, 둘째는 아래와 같은 일반적인 Word 편집기로 편집 가능한 `text` 포맷형태의 `*.sxt` 파일로 열기/저장하는 방식이다.

*.sxt 파일의 구조(예)

```
Symbol Generator Version 1
485 // Total Number of Symbol
```

```
HUD_A 9 // symbol Name & Total Number of Segment
INVISIBLE -24 -38 // Segment Type & FP x, y좌표
SOLID 0 60
SOLID 16 16
SOLID 16 0
SOLID 16 -16
SOLID 0 -60
INVISIBLE -48 38
SOLID 48 0
INVISIBLE 40 0
.
.
.
MFD_OP_15 9
SOLID 0 -744
CCW -184 0 184 0
SOLID 0 744
CCW 184 0 -184 0
SOLID -16 -20
SOLID 32 0
SOLID -16 20
INVISIBLE 184 0
INVISIBLE 0 -372
```

SYMBOL 선택/추가

SYMBOL LIST 대화상자에서 심볼 선택

<List> 메뉴는 아래 그림 9와 같이 기존에 작성된 모든 심볼들의 심볼명이 `Symbol List` 화면 내에 열거되게 하여 “심볼선택”, “새이름”, “찾기”, “잘라내기”, “붙이기”, “지우기”, “부른 매크로 보기”, “붙여진 매크로 보기”와 같은 부 메뉴로 수정/편집할 수 있도록 한다. 먼저 “심볼선택”은 `Symbol List` 화면으로부터 수정하고자하는 심볼을 선택한 후, `Graphic` 또는 `Text` 편집모드에서 수정/편집할 수 있도록 한다. “새이름”은 기존 심볼명을 변경 시 사용되며, “찾기”는 `Symbol List` 화면에서 심볼명을 쉽게 찾고자 할 때 사용된다. 즉, 찾고자 하는 심볼명이 `Symbol List`에 있으면 해당 심볼명으로 바로 이동하여 파란색 커서 바로 표시하도록 한다. 찾고자 하는 심볼명이 없는 경우에는 일치되는 심볼을 찾을 수 없다는 메시지를 띄운다. “잘라내기”, “붙이기”는 `Symbol List`에서 심볼들의 위치를 변경할 때 사용하며, “지우기”는 심볼을 `Symbol List`로부터 삭제한다. “붙여진 매크로 보기”는 `Symbol_List`에서 현재 선택한 심볼을 하위 매크로 심볼로 사용하고 있는 심볼들을 `ShowListDlg` 대화상자에 표시한다. “부른 매크로 보기”는 `Symbol_List`에서 현재 선택한 심볼에 대해 이 심볼을 구성하는 하위 매크로 심볼들의 심볼명을 `ShowListDlg` 대화상자에 표시한다.

그림 9. SYMBOL LIST 화면

SYMBOL NEW 대화상자에서 심볼 추가

`Symbol List`에 새로운 심볼을 추가하기 위해서는 초기화면에서 “NEW” 메뉴를 선택한다. “NEW” 메뉴가 선택되면 그림 10과 같은 `DlgNewName` 클래스 대화상자가 열리며 새로운 심볼명을 기입하면 `Symbol List`에 새로운 심볼명이 추가된다.

그림 10. NEW SYMBOL 추가화면

이때 기입되는 심볼명의 각 문자는 CheckValidityName(Cstring str) 함수를 사용하여 '0'-'9', 'A'-'Z', 및 '_'의 범위에 해당하는가를 검사하여 심볼명 내에 하나의 문자라도 상기 범위를 벗어나면 "Symbol Name is not Valid!" 메시지 박스를 띄운다. 또한 새로이 기입되는 심볼명이 Symbol List에 있는 기존 심볼명과 중복된다면 역시 "Symbol Name is not Valid!" 메시지 박스를 띄운다.

SYMBOL 편집

심볼을 편집할 수 있는 모드는 크게 Graphic 모드와 Text 모드가 있다. Graphic 모드는 grid가 표시된 화면에 직접 심볼을 그리면서 심볼을 편집할 수 모드이며, Text 모드는 두 개의 화면으로 구성되어 심볼의 전반적인 형상을 확인하면서, 하나의 심볼을 구성하고 있는 모든 segment(세그먼트)들의 특성 즉, SegmentType, First Point, Second Point, Macro Call등을 문자 형태로 편집할 수 있는 모드이다.

Graphic 모드

Symbol List 화면에서 심볼명을 선택한 후, "Graphic" 메뉴 선택 시에는 OnGraphic() 함수가 실행되어 선택된 심볼 객체로부터 심볼을 구성하는 총 segment 개수(m_nNoSegment 변수), 심볼의 시작점(m_pntSymbolStart 변수)과 끝점(m_pntSymbolEnd 변수)의 위치 값을 계산하고 Invalidate() 함수를 호출 및 OnDraw 이벤트를 발생시켜 OnDraw(CDC* pDC)를 실행시킨다. OnDraw(CDC* pDC) 함수는 m_modeDraw 열거형 변수에 의해 여러 가지 다양한 함수가 수행되는데, m_modeDraw 열거형 변수의 변수 값이 M_SAVE가 아닌 경우에는 DrawWholeSegment(pDC) 함수가 실행되어 선택된 심볼의 모든 segment를 SegmentType에 따라 CDC 클래스의 멤버함수인 MoveTo, Line, Arc 등을 이용하여 그린다. 아래 그림 11은 Symbol List 화면에서 FPM(Flight Pass Marker) 심볼명을 선택한 후, 수정하고자 "Graphic" 메뉴를 선택한 경우 보여

그림 11. Graphic 모드에 (FPM)

그림 12. 심볼 그리기 보조메뉴

지는 화면 예이다. 그러나 New Symbol 추가화면에서 새로운 심볼을 추가하는 경우에는 총 segment 개수(m_nNoSegment 변수)가 0이어서 segment 그리기 알고리즘이 실행되지 않고 DrawStartPoint(), DrawEndPoint(), DrawGrid(pDC) 함수만 수행되어 화면상에 StartPoint, EndPoint 및 grid만 그려진다.

Graphic 모드에 들어가면 심볼 그리기, 수정 및 저장에 위한 기호형태의 7가지의 부 메뉴 바가 활성화 된다. 메뉴, 메뉴 및 메뉴는 심볼을 그릴 때 사용되는 메뉴로써 각각 선폰, 원호 및 이미 구성되어 있는 심볼을 매크로로 불러서 사용하고자 할 때 사용되는 메뉴들이다. 또한, grid 크기 조정, segment 지우기 및 지우기 취소, 심볼 시작점/끝점 이동과 같은 심볼 그리기를 위한 보조 pop-up 메뉴가 그림 12와 같이 마우스 오른쪽 버튼을 누르면 화면상에 나타난다.

심볼 그리기용 3가지 메뉴와 보조 pop-up 메뉴를 사용하여 심볼을 화면상에 수정 또는 추가한 후, 심볼은 반드시 메뉴를 사용하여 저장되어야 한다.

메뉴를 선택하면 ExtractSolidSegment() 함수가 실행되면서, 수정 또는 추가된 심볼에서 INVISIBLE LINE을 제외한 SOLID segment에 대해서만 segment별로 끝단에 작은 원이 생성된다. 이때 사용자가 생성된 segment 끝단의 작은 원을 마우스로 순차적으로 선택하면 AddThisSegment() 함수가 실행되어 심볼을 구성하는 각 segment가 CSegment(CSegmentType SegmentType, CPoint pointFP, CPoint pointSP, CString strMacro) 객체에 저장된

다. 심볼의 모든 segment가 저장되면 자동으로 Graphic 모드를 빠져나가고, 새로이 수정 또는 추가된 심볼이 저장되었음을 메시지 박스로 알려준다. 또한 수정 또는 추가된 심볼에 대해 m_symbolList 데이터 구조 역시 수정 또는 추가된다.

Text 모드

Symbol List 화면에서 심볼명을 선택한 후, Text 메뉴를 선택하면 OnText() 함수가 실행되고 이 함수내의 dlg.DoModal() 함수에 의해서 Symbol Text Edit 대화상자가 그림 13과 같이 열린다.

이때, 대화상자의 좌측 edit(편집) 창에는 OnInitDialog() 함수가 실행되면서 선택된 심볼의 모든 segment가 SegmentType 별로 first point의 x, y 좌표 값 및 second point의 x, y 좌표 값을 표시한다. Symbol Text Edit 대화상자에는 Text 모드에서 심볼을 수정 또는 추가를 할 수 있도록 “저장하기”, “되살리기”, “그리기”, “오려두기”, “복사하기”, “붙이기”, “지우기” 등의 하위 메뉴가 있다. “그리기” 메뉴 선택 시에는 OnButtonDisplay() 함수 내 GetSizeSymbol(rectSize, pntCur, m_pSymbol) 함수가 실행되어 심볼 크기가 구해지고, 또한 GetClientRect(&rectGraphic) 함수가 실행되어 대화상자의 우측 display(시현) 창의 전체 크기가 계산된다. 계산된 크기를 이용하여 심볼 형상이 display 창안에 모두 그려지도록 심볼 크기를 일정비율로 축소시킨다. 따라서 display창에 그려진 심볼을 통해서 쉽게 심볼의 전체 형상을 파악할 수 있다. 한편, Text 모드 상에서 심볼 수정 및 추가 작업은 반드시 “저장하기” 명령이 실행되어야 심볼로서 저장된다. “지우기”, “되살리기”, “오려두기”, “복사하기”, “붙이기” 메뉴들은 CEdit 클래스 함수를 이용한 Text 편집 기능을 지원하기 위한 메뉴들이다.

그림 13. Text 모드 예 (FPM)

이진 파일 생성

"Graphic" 또는 "Text" 모드에서 모든 심볼 작성을 완료 후 초기화면에서 "Prom" 메뉴를 실행시키면 GP1 또는 GP2 그래픽 보드의 플래시 메모리 번지 \$1007 0000에서부터 1007 FFFF 영역까지, 즉 1 sector(1sector=256 block 또는 256*256 byte) 크기의 영역에 탑재될 이진 파일(*.xxx)과 심볼명을 정의한 ada 파일(sym_def_ada)이 생성되며 실행 완료 후, 아래 그림 14와 같은 메시지 박스를 띄워서 Prom 실행 완료 및 생성된 전체 심볼 개수를 보여준다.

그림 14. Prom 실행 화면

이진 파일(*.xxx)을 생성하기 위한 알고리즘은 다음과 같다. 먼저 fopen 함수, fprintf 함수 등을 이용하여 선택된 파일에 아래와 같은 알고리즘에 의해서 다운로드를 위한 header(헤더) 데이터 기록, 심볼 포인터의 저장주소 지정, 출력파일 초기화, 심볼 저장주소 위치 지정과 같은 전처리 과정을 실행한다.

이진 파일(*.xxx) 생성 알고리즘중 전처리 과정

```
FILE *fprom,*fada;
if((fprom=fopen(strPromFile,"w+"))==NULL)
{
    AfxMessageBox("PROM file open failed!");
    return;
}
if((fada=fopen(strAdaFile,"w+"))==NULL)
{
    fclose(fprom);
    AfxMessageBox("ADA file open failed!");
    return;
}

//다운로드를 위한 header 데이터 기록
fprintf(fprom, "00000000\n"); //&start_address,
fprintf(fprom, "10070000\n"); //code_start_address
fprintf(fprom, "00000100\n"); //&code_block_size
fprintf(fprom, "00000000\n"); //&data_start_address
fprintf(fprom, "00000000\n"); //&data_block_size
fprintf(fprom, "00000000\n"); //&rid_start_address
fgetpos(fprom,&pos);
```



```
//심볼 포인터 저장주소 지정
pointer_pos=pos+(fpos_t)((unsigned long)0xF800*2
+ 0xF80*2);

//출력파일 초기화
for(i=1; i<=4096; i++)
    fprintf(fprom,
"00000000000000000000000000000000\n");

//심볼 저장주소 지정
pos = pos + 0x80 + 8;
fsetpos(fprom, &pos);
var_ptr = 0x40;
```

이후, 각 심볼별로 심볼을 구성하고 있는 모든 segment에 대하여 SegmentType 따라 다음과 같이 메인 처리 과정을 수행한다.

1) segment가 INVISIBLE인 경우에는 Invisible임을 표시하기 위하여 16비트 data 변수의 15, 14비트에 각각 1과 0을 넣은 후, 0x3FFF와 segment의 x 좌표 값(m_pointFP.x)을 AND 한 값과 다시 OR하여 data 변수에 기록한다. 이 data 값을 WriteData 함수를 이용하여 이진 파일에 기록한다. 이때, 1 line에 8개씩의 data가 쓰여지며 8개가 넘으면 line이 바뀌어 쓰여진다. segment의 y 좌표 값은 그대로 data에 저장 후, Symbol Table의 DATA영역에 쓰여진다.

2) segment가 SOLID인 경우에는 solid임을 표시하기 위하여 data변수의 15, 14비트에 각각 0과 1을 넣은 후, 0x3FFF와 segment의 x 좌표 값(m_pointFP.x)을 AND 한 값과 다시 OR하여 data에 쓴다. 이후 INVISIBLE과 동일한 처리를 거친다.

3) CW segment인 경우에는 arc임을 표시하기 위하여 data 변수의 15, 14, 13, 12, 11비트에 각각 1을 넣고 다시 CW임을 표시하기 위하여 최하위 비트를 1로 한다. 즉 16진수로 0xF801을 만들어서 data변수에 저장한다. 이후, arc(원호)의 x_start 좌표 값, y_start 좌표 값, x_end 좌표 값, y_end 좌표 값을 data에 순서적으로 저장 후, Symbol Table의 DATA영역에 쓴다.

4) CCW segment인 경우에는 arc임을 표시하기 위하여 15, 14, 13, 12, 11비트에 각각 1을 넣고 다시 CCW임을 표시하기 위하여 최하위 비트를 0으로 한다. 즉 16진수로 0xF800을 만들어서 data에 저장한다. 이후, CW segment 처리과정과 동일하다.

5) MACRO segment인 경우에는 MACRO임을 표시하기 위한 0xC000 와 몇 번째 심볼 인가를 나타내는 k값을 OR하여 data에 저장 후, Symbol Table의 DATA영역에 쓴다.

하나의 심볼에 대해 상기와 같은 메인 처리과정이 완료된 후에는 0xEB00을 이진 파일에 써서 하나의 심볼 처리가 완료되었음을 표시한다. 또한, fsetpos(fprom, &pointer_pos) 함수를 사용하여 Symbol Table의 LOOK-UP 영역의 시작위치로 이동하여 심볼의 시작번지를 저장한다. 이와 같은 과정을 m_symbolList 변수에 저장된 모든 심볼에 대해 반복 수행하여, Symbol Table 영역에 다운로드 될 이진 파일을 생성한다.

이진 파일 다운로드

“Prom” 메뉴를 실행하여 생성된 이진 파일 데이터는 아래 그림 15와 같이 PC의 RS 232 통신포트와 ASC의 GP보드에 내장되어 있는 controller(Z85230칩)를 통해 비동기 통신방식으로 전송되며, 전송된 데이터는 GP보드 플래시 메모리 영역 중에 심볼 테이블로 할당된 영역에 다운로드 된다.

“Drom” 메뉴를 선택하면 OnDrom() 함수가 실행되고, RS 232 비동기 통신을 위한 baudrate, 통신포트 등이 초기화된 다운로드용 대화상자가 열린다. 이후 “FileSelect” 메뉴로 다운로드 할 이진 파일(*.xxx)을 선택한다. 선택된 파일 데이터를 다운로드하기 위해서 “Upload” 메뉴를 선택하고 이에 따라 ProgUpload() 함수가 실행되면서 CCommThread 클래스 함수인 OpenPort()에 의해서 RS 232 통신포트가 열린다. 또한, fopen, fscanf 함수로 다운로드 할 이진 파일을 열고 header data가 순차적으로 읽혀서 ASC로 전송된다. header data 전송 후, Symbol Table 영역에 저장될 256 block(1block=256byte) 데이터가 전송된다. 이때 PC와 ASC 사이에 데이터를 송/수신하는 방식은 소프트웨어 핸드 셰이킹 방식으로 PC와 ASC 간의 송수신 준비상태를 점검하고 나서 헤더 데이터와 블록 데이터를 송신하는 방식

그림 15. Symbol Table 다운로드 개념도

을 사용한다[3]. Header data와 블록 데이터 전송 완료 후 fclose 함수로 파일을 닫고, CCommThread 클래스 함수인 ClosePort()에 의해서 RS 232 통신포트를 닫는다.

2.5 실제 시험기에 구현된 심볼형상

본 개발과제에서는 심볼 생성기용 소프트웨어인 SYMBOL-GEN을 사용하여 HUD용 심볼 183개와 MFD용 심볼 302개가 생성하였다. HUD용 심볼들로는 그림 16 HUD(상) 심볼 전시 화면의 중앙 상단에 보이는 FPM(Flight Pass Mark), BORESIGHT_CROSS, PITCH_BAR등과 같이 항공기의 진행방향, 자세 등에 관한 정보를 제공하는 심볼들과 좌/우측면에 보이는 고도, 속도 등과 같은 비행에 필수적인 문자, 숫자 심볼들이 생성되었다. 또한 MFD용 심볼들로는 그림 16 MFD(하) 심볼 전시 화면에서 보이는 HSI(Heading Situation Indicator) 심볼 등과 같이 MFD가 시현하여야 할 각종 화면에 사용되는 심볼들이 생성되었다. 생성된 심볼들은 다시 SYMBOL-GEN S/W로 이진 파일과 sym_def_ada파일로 각각 변환된 후, 이진 파일은 ASC의 그래픽 보드 플래시 메모

리내 Symbol Table 영역에 다운로드 되었고, sym_def_ada 파일은 개발된 OFP 프로그램과 함께 패키지 되어 실제 항공기에 탑재되었다. 탑재 후 실제 수 차례의 비행시험을 수행한 결과, SYMBOL-GEN에 의해서 생성/다운로드 된 모든 심볼들이 정상적으로 HUD 및 MFD 화면에 시현됨이 확인되었다. 그림 16은 실제 ASC의 항공기 탑재 비행시험 중에 HUD와 MFD상에 각각 시현된 심볼 전시 화면의 일례이다.

III. 결 론

본 논문은 군용항공기용 시험기인 HUD 및 MFD에 그려지는 여러 가지 심볼 형상들을 off-line에서 보다 효율적으로 생성/편집하여 ASC의 그래픽 보드내 플래시 메모리의 일정영역에 직접 다운로드 할 수 있는 S/W인 SYMBOL-GEN의 구성, 상세 개발내용 및 실제 적용결과를 기술하였다. SYMBOL-GEN은 ASC 운용프로그램인 OFP에서 직접 심볼을 그릴 때 발생 할 수 있는 그래픽 속도 저하 문제를 해결하였을 뿐만 아니라, 조종사의 다양한 심볼 형상 수정 및 추가 요구에 빠르고 효율적으로 심볼을 변경, 적용 할 수 있을 것으로 기대된다. 또한 객체 지향적인 S/W 설계로 필요시 용이한 수정 및 확장성이 보장되어 향후, 심볼 확대/축소 기능 등과 같은 편리한 기능들을 간단하게 추가/보완할 수 있을 것이라 판단된다.

참고문헌

1. 황상현, "항전시스템 컴퓨터용 그래픽 펌웨어 설계연구", 국방과학연구소 연구보고서, 2001.12
2. 황상현, 이재억, 박덕배 "전방시험기 및 다기능시험기 구동을 위한 그래픽 영상생성기 설계연구", 한국군사과학기술학회지, 2002.7
3. 이재억, 양승열, "항공전자 시스템 컴퓨터 CPU초기화 및 업로드/다운로드 프로그램 설계, 국방과학연구소 연구보고서, 2002.5
4. 이상엽, "Visual C++ Programming Bible Ver 6.0, 영진 출판사, 서울, 1999.

그림 16. HUD(상), MFD(하) 심볼전시화면