LEON3 프로세서 모니터링 소프트웨어 개발

류상문*

*군산대학교

Development of monitoring software for LEON3 processor

Sang-Moon Ryu*

*Kunsan National University

E-mail : smryu@kunsan.ac.kr

요 약

LEON3는 SPARC V8을 기반으로 구현된 32비트 마이크로프로세서이다. 7 단계 파이프라인, IEEE-754 FPU 그리고 256[KB] 캐쉬 등을 지원하며 AMBA 2.0 버스에 접속될 수 있다. LEON3는 합성 가능한 VHDL로 기술되어 있어 FPGA로 구현하기 용이하며 SoC 설계에도 사용할 수 있다. LEON3와 함께 제공되는 DSU를 AMBA 버스를 통하여 접근하면 LEON3의 동작을 제어하거나 동작 상태를 파악할 수 있으며, 이를 이용하여 LEON3를 기반으로 동작하는 임베디드시스템의 하드웨어와 소프트웨어를 개발하거나 디버깅할 수 있는 환경을 갖출 수 있다. 본 논문은 DSU를 이용하여 LEON3의 동작을 통제하고 그 상태를 파악할 수 있는 LEON3 모니터링 소프트웨어의 개발 결과를 정리한 것이다.

ABSTRACT

LEON3 is a 32-bit synthesisable processor based on the SPARC V8. It can be connected to AMBA 2.0 bus and has a 7-stage pipeline, IEEE-754 FPU and 256[KB] cache. It can be easily implemented using FPGA and used for a SoC design. DSU which comes with LEON3 can be used to control and monitor the operation of LEON3. And DSU makes it easy to set a debugging environment for the development of both hardware and software for an embedded systems based on LEON3. This paper presents the summary of the development of LEON3 monitoring software.

키워드

LEON3, 임베디드시스템, 디버깅

Ⅰ. 서 론

FPGA의 집적도가 높아짐에 따라 FPGA를 이용하여 SoC 기반의 임베디드시스템 하드웨어를 개발하는 것이 매우 용이해졌다. 고속의 연산 처리가 필요한 경우에는 프로세서 하드코어가 내장된 FPGA를 사용하고 그렇지 않은 경우에는 IP 형태로 제공되는 ARM사의 Cortex-M1, Microsemi사의 Core8051, Xilinx사의 MicroBlaze 또는 Gaisler사의 LEON 같은 소프트 프로세서를 사용할 수 있다.

특히 우주 환경에 적합한 RT(Radiation Tolerant)급 부품을 사용하여야 하는 인공위성 같은 우주 개발 프로그램에서는 유럽을 중심으로 LEON 같은 소프트 프로세서를 RT급 FPGA나 ASIC으로 구현하여 사용하고 있는 추세이며, 국내에서도 마찬가지 방법으로 진행되고 있다. 이것은 미국의 방산 물자 수출 제한(Export License) 규정 때문에 미국산 RT급 마이크로프로세서 부품을 유럽이나 국내에서 사용할 수 없기 때문이다[1][2][3].

LEON3와 함께 제공되는 DSU(Debug Support Unit)[4]를 AMBA 버스를 통하여 접근하면 LEON3의 동작을 제어하거나 동작 상태를 파악할

수 있으며, 이를 이용하여 LEON3를 기반으로 동작하는 임베디드시스템의 하드웨어와 소프트웨어를 개발하거나 디버깅할 수 있는 환경을 갖출 수 있다[5].

임베디드시스템의 모니터링 도구는 하드웨어나 소프트웨어 개발에 필수 불가결한 요소이며 채용되는 마이크로프로세서에 매우 종속적이다. 본 논문은 DSU를 이용하여 LEON3의 동작을 통제하고 그 상태를 파악할 수 있는 LEON3 모니터링 소프트웨어의 개발 결과를 정리한 것이다.

II. 본 론

2.1 모니터링 소프트웨어의 구성

본론 그림 1은 개발된 LEON3 모니터링 소프트웨어가 DSU를 통하여 LEON3 기반 임베디드 시스템 하드웨어를 통제하고 그 상태를 파악하는 과정을 보여주기 위한 그림이다. LEON3 디버깅 도구는 RS-232 직렬 통신을 통하여 Serial Debug I/F와 통신하고 AMBA 버스의 AHB Master I/F를 갖고 있는 Serial Debug I/F는 요청에 따라 AMBA 버스에 읽기 또는 쓰기 동작을 수행한다. LEON3 프로세서의 내부에 접근하고자 할 때에는 DSU에 접근하고, 메모리에 접근하고자 할 때에는 Memory Controller에 접근한다. 따라서 LEON3 모니터링 소프트웨어가 AMBA 버스의 마스터로서 버스에 접속된 DSU와 메모리에 직접 접근할 수 있다는 것이다.

DSU는 AMBA 버스에서 일어난 최근 동작을 저장하는 AHB TB(Trace Buffer)에 직접 접근할 수 있으며 Debug I/F를 통하여 LEON3 프로세서의 FPU(Floating Point Unit), 각종 내부 레지스터, Instruction 및 Data Cache 등에 접근할 수 있다.

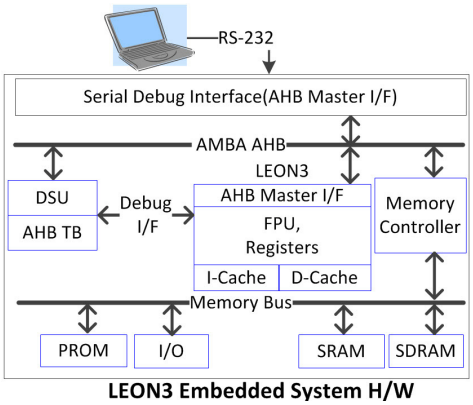

그림 1 모니터링 S/W 동작 개요

개발된 LEON3 모니터링 소프트웨어의 구성 블록도는 그림 2와 같다. 타겟 하드웨어 초기화 블록은 타겟 하드웨어의 AMBA 버스에 접속된 주변 장치들의 레지스터들을 사용자가 설정한 값들로 설정하고, 특정 주소에 저장되어 있는 플러그 앤 플레이 정보를 읽어서 타겟 하드웨어를 구성하는 IP 목록과 할당된 주소 목록을 획득하여 타겟 하드웨어의 구성 정보를 수집한다.

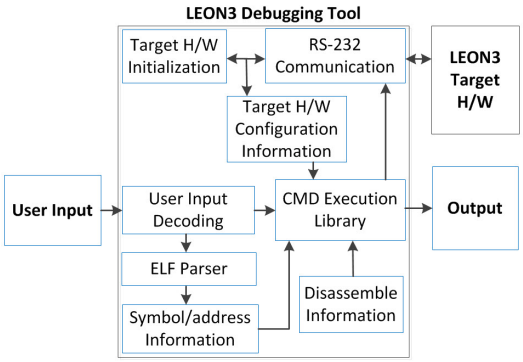

그림 2 LEON3 모니터링 소프트웨어 구성도

ELF(Executable and Linkable Format) 파일 구문분석기(parser)가 포함되어 있어 사용자에 의해 선택된 ELF 파일을 분석하여 메모리에 적재해야 할 실행 코드를 추출하고, 전역 변수와 함수 등의 심볼과 해당 주소 등에 관한 정보를 수집한다.

그리고 타겟 하드웨어의 메모리에 적재된 실행 코드의 2진 코드를 SPARC V8의 명령어 집합(instruction set) 정보를 바탕으로 어셈블리 언어로 변환하는 역어셈블러(disassembler)가 포함되어 있다.

사용자가 디버깅을 위해 입력한 명령은 명령 수행 라이브러리(CMD execution library) 블록에서 타겟 하드웨어의 구성 정보, ELF 파일에 포함된 심볼과 주소 변환 정보, SPARC V8 역어셈블 정보 등을 참조하여 처리되고 RS-232 직렬 통신 블록을 통하여 타겟 하드웨어에 전달된다. 그리고 타겟 하드웨어에서의 응답이 해석되어 문자열 형태로 변환되어 출력된다. 그림3은 개발된 LEON3 모니터링 소프트웨어의 시험 및 검증을 위해 구성된 실행 화면이다.

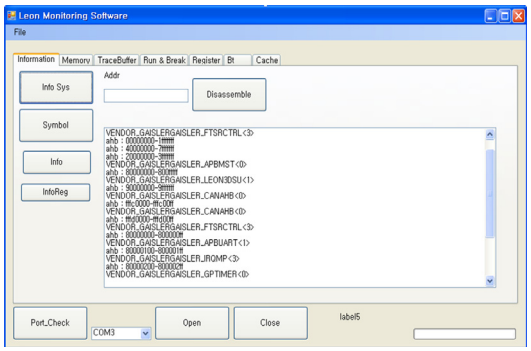

그림 3 LEON3 모니터링 S/W 시험 화면

2.2 타겟 초기화

타겟 하드웨어의 AHB 버스와 APB 버스에 접속되어 있는 주변 장치의 목록과 각 주변 장치에 할당된 주소를 파악하여 이들을 사용자의 설정에 따라 초기화한다. 주변 장치의 목록과 각 주변 장치에 할당된 주소는 각 장치별로 특정 주소에 저장되어 있는 Plug&Play 정보를 이용하여 파악할 수 있다(그림 4).

```
VENDOR_GAISLERGAISLER_FTSRCTRL<3>
ahb : 00000000-1fffffff
ahb : 40000000-7fffffff
ahb : 20000000-3fffffff
VENDOR_GAISLERGAISLER_APBMST<0>
ahb : 80000000-800fffff
VENDOR_GAISLERGAISLER_LEON3DSU<1>
ahb : 90000000-9fffffff
VENDOR_GAISLERGAISLER_CANAHB<0>
ahb : fffc0000-fffc00ff
VENDOR_GAISLERGAISLER_CANAHB<0>
ahb : fffd0000-fffd00ff
VENDOR_GAISLERGAISLER_FTSRCTRL<3>
ahb : 80000000-800000ff
VENDOR_GAISLERGAISLER_APBUART<1>
ahb : 80000100-800001ff
VENDOR_GAISLERGAISLER_IRQMP<3>
ahb : 80000200-800002ff
VENDOR_GAISLERGAISLER_GPTIMER<0>
```

그림 4 주변 장치 목록 및 할당 주소 출력 예

2.3 실행 코드 적재 및 실행

실행 코드 적재 기능은 2진 코드 파일과 ELF 파일을 지원한다. 2진 코드 파일은 사용자가 지정한 메모리의 주소 영역에 적재하고, ELF 파일의 경우에는 ELF 파일 내부를 분석하여 실행 코드가 적재되어야하는 메모리의 물리 주소를 획득하고, PT_LOAD 세그먼트를 찾아 메모리에 적재되어야할 실행 코드를 확보한다.

실행 코드 적재가 완료되면 LEON3 프로세서가 이를 수행할 수 있도록 LEON3 프로세서의 PC(Program Counter), NPC(Next Program Counter), SP(Stack Pointer), WIM(Window Invalid Mask) 그리고 PSR(Processor Status Register) 등을 적정값으로 설정한다(그림 5).

```
Run
Program Exited Nomarly : 80
PC : 0x40000800 : Data : 91d02000 dis :ta  ,0x0
```

그림 5 어플리케이션 실행 예

2.4 중단점(Break Point) 설정 및 적용

중단점은 사용자가 원하는 위치에서 Trap을 발생하여 LEON3 프로세서의 동작을 일시 중지시키고 레지스터나 메모리에 저장되어 있는 값을 파악하거나, 최근에 호출된 함수의 목록을 얻을 수 있도록 해준다. 중단점은 소프트웨어적인 방법, 하드웨어적인 방법으로 삽입할 수 있다.

소프트웨어적인 중단점은 사용자가 원하는 메모리 물리 주소에 Trap을 발생하는 'ta 1' 명령을 기입하여 삽입할 수 있다. 이를 위해 'ta 1' 명령이 기입되는 지점에 이미 저장되어 있던 명령어를 읽어 놓았다가 중단점 적용 이후 복원하여 주어야한다. 그리고 중단점이 삽입되어야하는 메모리 영역이 인스트럭션 캐시에 존재하는지 파악하여 필요시 인스트럭션 캐시에서 중단점에 해당하는 지점에 에도 'ta 1' 명령을 기입하여 준다.

하드웨어적인 중단점은 LEON3 프로세서의 IU(Integer Unit)에 포함되어 있는 ASR(Ancillary State Register)를 이용하여 삽입할 수 있다. 중단점을 삽입하고자 하는 메모리 물리 주소를 ASR에 저장하고 특정 비트들을 적절히 설정하면 LEON3 프로세서가 중단점이 설정된 메모리 지점에 접근할 때 Trap이 발생된다. 그림 6은 설정된 중단점 목록을 보여준다.

중단점에 의해 Trap이 발생한 LEON3 프로세서는 DSU에 포함된 Break Single Step 레지스터를 이용하면 명령어 단위로 그 실행을 통제할 수 있다.

```
Bus Watch Point Limited : 2
Break Point
num : 0    Addr : 0x40000000 Mode : Soft
num : 1    Addr : 0x40000004 Mode : Hard
num : 2    Addr : 0x40000008 Mode : Bus Watch
num : 3    Addr : 0x4000000c Mode : Watch
num : 4    Addr : 0x4000162c Mode : Soft
num : 5    Addr : 0x4000155c Mode : Soft
num : 6    Addr : 0x4000159c Mode : Soft
num : 7    Addr : 0x4000158c Mode : Soft
num : 8    Addr : 0x4000158c Mode : Bus Watch
```

그림 6 중단점 설정 예

25 Back Trace 출력

Back Trace는 LEON3 프로세서의 동작이 중단점에서 일시 중단된 상태에서 최근에 호출된 함수 목록을 출력하는 기능이다. LEON3 프로세서의 레지스터에 남아 있는 최근 함수 호출 흔적을 WIM(Window Invalid Mask)와 CWP(Current Window Pointer) 값을 참고하여 해석하고, 이를 ELF 파일의 구문 분석을 통해서 획득한 함수-메모리 주소 정보와 대조하여 최근에 호출된 함수 목록을 각 함수 호출 당시의 PC와 SP 값과 함께 출력한다(그림 7).

```
PC : 40000800      SP : 407ffea0      start+0x800
PC : 40001a58      SP : 407ffea0      main+0x42c
PC : 4000106c      SP : 407fff60      _start+0x6c
PC : 4000a41c      SP : 407fffa0      _hardreset_real+0x80
```

그림 7 Back Trace 출력 예

III. 결 론

우주개발 프로그램에 사용할 수 있는 전자 부품은 일반 제품 제작에 사용되는 전자 부품과 다르게 그 종류나 수급이 상당히 제한된다. 그 중 가장 엄격히 제한되는 것이 마이크로프로세서이다. 유럽은 이를 극복하기 위해 SPARC V7을 기반으로 LEON3를 개발했으며 이를 우주 복사 환

경에서도 동작할 수 있는 LEON3FT(Fault Tolerance)로 구현하여 우주 개발 프로그램에 적용하고 있다. 이미 LEON3의 다음 버전인 LEON4도 개발되었다. 우리나라의 우주 개발 프로그램에서도 LEON3FT가 사용되고 있다.

LEON3는 GNU 라이센스에 따라 무료로 연구나 교육용으로 사용할 수 있다. 본 연구에서 개발된 디버깅 도구를 활용하면 교육 및 연구 목적의 LEON3 수정이나 성능 개선이 효율적으로 진행될 수 있을 것이라 기대된다. 특히 LEON3와 AMBA 버스를 기반으로 임베디드시스템 하드웨어를 개발할 때에도 유용하게 활용될 수 있을 것이라 기대된다.

참고문헌

[1] 최종욱, 천이진, “탑재소프트웨어 개발 측면에서의 차세대 위성탑재컴퓨터의 프로세서 분석”, 한국항공우주학회 2012년도 추계학술대회논문집, pp.809-814, 2012.

[2] 오대수, 박성옥, 강경인, 명로훈, “나로 과학위성의 탑재컴퓨터 설계 및 구현”, 한국항공우주학회 2011년도 추계학술대회논문집, pp.1437-1442, 2011.

[3] 이선재, 류정환, 박준용, 정성근, 우형제, 조영호, 원주호, “정지궤도인공위성을 위한 탑재컴퓨터 설계”, 한국항공우주학회 2009년도 추계학술발표회 논문집, pp.990-993, 2009.

[4] J. Gaisler, A. Pouponnot, “Hardware debug support in the LEON processor”, Proceedings of DASIA 2002, p.1-5, May 2002.

[5] LEON3-FT SPARC V8 Processor Data Sheet and User’s manual, Aeroflex Gaisler, 2012.