블루투스 임베디드 시스템을 위한 GPIO 설계

문상국*

Design of a GPIO Unit for Bluetooth Embedded Systems

Sangook Moon*

요 약

본 연구에서는 임베디드 시스템, 특히 블루투스 베이스밴드에서 사용이 가능한 범용 GPIO (general purpose input output)를 설계하였다. 제안하는 구조는 AMBA 버스구조의 APB 저전력 버스에 호환되도록 설계하였다. 응용 분야가 다양한 임베디드 시스템에서는 GPIO 방식의 인터럽트 소스가 가장 유용하게 사용된다. 본 논문에서는 에지 반응 방식과 레벨 반응 방식 모두를 고려하여 인터럽트를 수용할 수 있도록 설계하였고, 동작 폴라리티를 선택할 수 있어 다양한 응용의 블루투스 임베디드 디바이스에 유연하게 적용될 수 있도록 고려하였다. 설계한 GPIO 모듈은 Altera QuartusII 툴을 사용하여 자동합성하여 P&R을 수행하였다. 결과물은 CycloneII FPGA로 구현하였으며 타겟으로 정한 25MHz에서 충분히 동작 가능하다.

ABSTRACT

In this contribution, we designed a general purpose input/output (GPIO) suitable for embedded systems, especially for Bluetooth baseband. Proposed architecture is compatible for the APB bus in AMBA bus architecture. General purpose I/O should be used as multi-functional and versatile interrupt sources. We considered the edge-sensitive mode as well as the level-sensitive mode for acquiring the interrupt sources. Also, we provided an option to select the operation polarity for flexible application to the embedded systems. The designed GPIO module was automatically synthesized, placed, and routed. The proposed GPIO was implemented through the Altera FPGA and well operated at 25MHz clock frequency.

키워드

임베디드시스템, 베이스밴드, GPIO

Key word

Embedded Systems, Baseband, GPIO

* 정회원 : 목원대학교(smoon@mokwon.ac.kr)

접수일자 : 2011. 08. 25
심사완료일자 : 2011. 11. 16

Ⅰ. 서 론

블루투스 기술은 작고 저렴한 가격, 저전력 소모로 근거리 송수신기를 소형 이동형 디바이스에 직접 또는 어댑터를 통해 탑재되어 무선 환경을 제공하는 무선 통신 규약 중 하나이며, IEEE 802.15에 표준화되어 있으며 현재 버전 4.0까지 발전하였다 [1]. 무선 환경은 세계적으로 이용이 가능한 전역 주파수 대역인 2.4Ghz 밴드를 이용하고 모드에 따라 약 700kbps 까지의 전송 속도를 지원하며 3개의 음성 채널을 지원한다. 또한 이동성을 고려하여 다양한 전력 소비 모드도 가지고 있다.

도달 거리에 따라서도 지원 가능한 거리를 나누어 각각 다른 모드로 동작할 수 있다 [2]. 1994년 에릭슨의 이동통신그룹에서 휴대폰과 주변기기들간의 소비전력이 적고 가격이 저렴한 무선 인터페이스를 연구하기 시작한 것이 그 시초이다. 현재 블루투스 SIG에 참여하고 있는 회사들은 모토롤라, 마이크로소프트, 루슨트테크놀로지, 3COM과 설립 그룹인 에릭슨, 노키아, IBM, 도시바로 구성되어 있다 [3].

블루투스의 구조는 물리층을 규정하는 RF, 호핑 패턴 및 통신 제어를 담당하는 베이스밴드, 패킷의 구성을 규정하는 링크 매니저, L2CAP과 상위 호스트간의 인터페이스를 규정하는 HID와 RFCOMM으로 나누어진다 [4].

베이스밴드는 AMBA 버스 아키텍쳐 기반으로 구성되었으며, 슬레이브 모듈 중 하나로 GPIO가 구성된다. GPIO는 속도에 민감하지 않기 때문에 APB 부분에 구현하는 것이 일반적이며 세팅에 따라 인터럽트 소스로 동작할 수 있어야 한다 [5]. 기존의 인터럽트 제어기가 양의 값에 기반한 (active-high) 신호에 의해서만 제어되기에 유연성이 떨어지는 것에 비해 GPIO는 다수의 제어 레지스터를 설정하여 자유롭게 구현할 수 있는 장점이 있어 입력 극성, 에지-레벨 선택을 가능하게 구현할 수 있다. 본 논문에서는 유연한 방식의 GPIO를 FPGA로 구현하여 칩 레벨의 포스트-레이아웃 레벨에서 시뮬레이션하여 정상적인 동작을 검증하였다.

Ⅱ. 블루투스 베이스밴드

그림 1은 블루투스 베이스밴드의 일반적인 블록 다이어그램이다. 베이스밴드의 세부 블록은 아래와 같이 세분화된다.

- RF 인터페이스
- LMP 인터페이스
- 홉 선택 제어기
- ACL & SCO 링크 제어기
- HEC, FEC 제어기
- 액세스코드 상관기
- 데이터 화이트닝
- 암호화
- 인증
- 오디오 인터페이스
- 클럭 복원과 동기화 모듈
- 채널 제어기
- RX/TX 버퍼와 레지스터

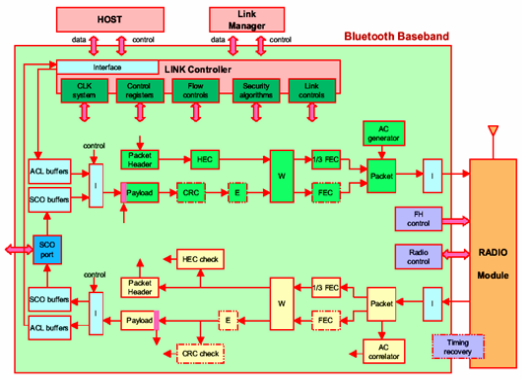

그림 1. 블루투스 베이스밴드 블록 다이어그램
Fig. 1 Bluetooth baseband block diagram

그림 오른쪽에는 RF 모듈이 존재하여서 2.4 GHz ISM 밴드의 주파수를 변조하는 역할을 수행하고, 이로 받은 데이터들은 아래 부분의 타이밍 복원 회로에 의해 1MHz의 샘플링 형식에 맞는 데이터의 스트림으로 입력된다. 입력된 데이터들은 먼저 로우패스 필터를 통과하여 노

이즈를 제거하면서 신호의 수행 사이클에 충실히 번역되어 베이스밴드 내로 전달된다.

이 때 64비트 블록 단위로 전송되는 데이터는 데이터의 싱크 검색기에 의해 신호의 문자열이 '1010' 또는 '0101'인지를 감별하여 그것이 데이터 패킷의 시작인지를 인지한다. 성공적으로 인지되어 전달된 데이터는 일정한 블록 형태의 병렬 데이터로 변환되고 변환된 데이터는 헤더와 페이로드 부분으로 나뉘어 각각의 에러 정정 블록으로 전달된다. 헤더 에러 검출 블록에서는 스펙에 정의된 헤더의 에러를 검출하는 알고리즘을 이용하여 헤더의 에러를 검출하여 이후를 진행하고, 페이로드 부분은 화이트닝/디화이트닝 블록에 의해 스크램블링 되었던 데이터들이 의미있는 값을 가지면서 FEC (Forward Error Correction) (1/3, 2/3) 에 의하여 에러를 다시 검출하고, [6][7] 간단한 에러 복구 메커니즘에 의해 복구된 데이터는 헤더와 페이로드로 분리되어 베이스밴드 내 메모리에 데이터 블록으로 각각 저장된다.

통신 패킷의 종류는 SCO (Synchronous Connection Oriented)와 ACL (Asynchronous Connection Oriented) 두 종류로 나뉜다. SCO 패킷은 슬롯을 할당하여 주로 음성통신과 같은 어플리케이션에 사용되고, ACL은 유연성을 가지는 데이터 패킷들을 전송하는데 주로 사용된다 [8].

Ⅲ. 블루투스 임베디드 시스템과 GPIO

블루투스 시스템은 다양한 응용에 사용될 수 있으며 가장 주요하게 쓰이는 예는 무선 헤드셋, 무선 스피커, 무선 게임 콘트롤러 등이 해당된다. 이런 디바이스들은 공통적으로 사용자와 버튼으로 의사 소통을 전달하며, 이러한 버튼으로 구현 가능한 인터페이스가 GPIO이다.

그림 2는 타겟 블루투스 임베디드 시스템의 전체적인 구조이다. 전체 시스템은 여러 IP들이 AMBA (advanced microcontroller bus architecture) ASB (advanced system bus), FastAPB (advanced peripheral bus), SlowAPB 버스로 연결되어 있는 구조이다.

AMBA ASB에 연결된 IP들은ARM7TDMI 마이크로프로세서 코어가 포함된 메모리와 핀 멀티플렉서, 테스트용 포트로 이루어져 있으며 시스템 IP들을 포함한다. 이와 연결된 AMBA FastAPB 버스와 연결된 IP들은 USB와 타겟 임베디드시스템의 핵심이 되는 베이스밴드 코어가 있다. 또한 AMBA SlowAPB 버스와 연결된 모듈들은 시간적인 제약을 크게 받지 않거나 데이터 처리속도가 비교적 느린 시리얼 인터페이스와 타이머 등을 포함한다.

이 중 본 연구에서 설계한 부분은 AMBA SlowAPB 버스로 연결한 주변 모듈의 하나인 GPIO이다 (그림 2 음영부분). GPIO는 ASB 영역의 모듈과는 동작 속도의 차이가 발생하기 때문에, 타이머나 UART와 함께 SlowAPB 영역에 할당이 되어 있다. GPIO가 프로세서 코어가 존재하는 ASB 부분과의 속도를 맞추기 위하여 SlowAPB Bridge 모듈이 존재하며, 이를 통해서 고속 ASB 시스템과 통신한다. APB Bridge는 비교적 느린 APB IP들과 속도가 빠른 ASB 시스템 IP들의 서로 다른 속도를 가진 IP들을 이어주는 다리와 같은 역할을 하며, 시스템 IP로부터 받은 ASB 프로토콜을 APB 프로토콜로 변환하여 APB IP에 전달하거나 APB 프로토콜을 ASB 프로토콜로 변환하여 시스템으로 데이터를 전달한다.

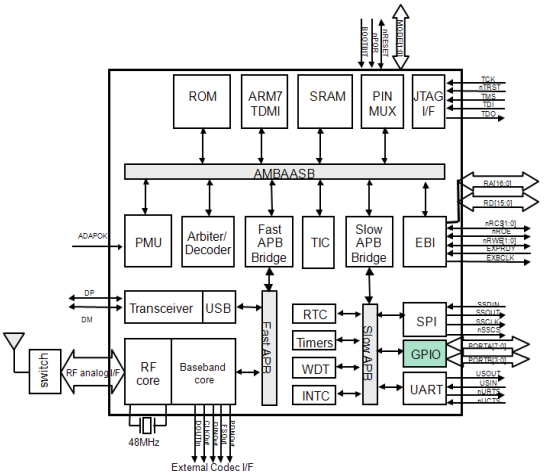

그림 2. 타겟 블루투스 임베디드 시스템
Fig. 2 Target bluetooth embedded system

Ⅳ. 2 포트 GPIO

하나의 GPIO 단자를 설계하려면 GPIO 레지스터, 에지 검출 회로, 멀티플렉서, 3상태 버퍼 등이 필요하다. 본 논문에서 설계한 GPIO는 하나의 연결 단자를 입력이나 출력으로 고정시키지 않고 선택적으로 사용할 수 있도록 하기 위하여 포트 당 8비트 외부 입력, 외부 출력과 외부 입력에 의한 인터럽트 신호를 출력하며, 추가적으로 내부 마이크로프로세서와 통신을 위한 AMBA 기반의 8비트 양방향 입출력 버스를 가진다. 또한, 소비전력을 줄이기 위하여 포트 개념을 도입하여 실제 필요한 포트만 사용할 수 있도록 설계하였다. 전체적인 구조는 A, B, 두 포트로 이루어진 그림 3과 같다.

PA, PD, PWRITE, PSELGPIO, PSTB의 신호들이 AMBA APB 프로토콜에 기반한 신호이며, 표 1에 보인 기능 레지스터에 값을 각각 저장하거나, 기능 레지스터의 값을 읽기 위해 사용된다. 외부 입력 포트로 PORTAExtIn, PORTBExtIn을 두어 외부 패드를 입력장치로 쓸 때 입력 신호를 받을 수 있도록 하였고, GPIO 패드를 출력장치로 사용할 때는 그림 4의 DataR 레지스터의 값을 APB 프로토콜로 조작하여 각각 PORTAExtout, PORTBExtout 출력으로 내보내도록 하였다. 외부 패드를 입력으로 할 것인지 출력으로 할 것인지는 그림 4의 DirR 레지스터에 값을 넣어 외부 패드에 존재하는 3상 버퍼의 제어 신호로 보낸다.

그림 4에는 표 1의 레지스터의 동작을 도식적으로 나타내었다. MaskR 레지스터는 GPIO가 외부 입력으로 작용할 시 입력을 사용자의 요구에 따라 무시할 수 있는 역할을 하고, EdgeR 레지스터의 값에 따라 외부 패드의 입력 인터럽트를 눌리는 즉시 받아들일 것인지, 아니면 레벨 값을 저장하여 사용할 것인지를 결정한다. 외부 패드 입력으로 받아들인 데이터는 ClearR 레지스터의 값에 의해 비트별로 클리어 할 수 있고, PolarR 레지스터를 두어 외부 입력의 인터럽트 값을 active high로 할 것인지 아니면 active low로 할 것인지 결정할 수 있도록 설계하였다.

표 1은 GPIO의 주요 역할을 수행하는 레지스터에 대한 크기와 메모리 맵에서의 위치, 읽기/쓰기 권한, 기본 값을 나타낸다. 베이스 주소는 0x8002_3000이며 실제 사용하는 데이터는 8비트이지만 추후 확장성을 위해 32비트 레지스터를 할당하였다. 표에 나타나 있는 레지스터의 이름이 기능을 의미한다.

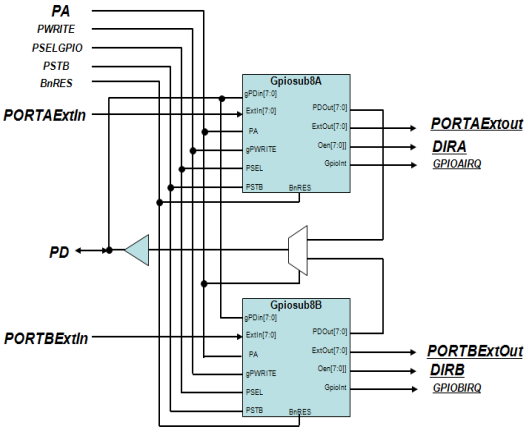

그림 3. GPIO APB 프로토콜 레벨 블록도
Fig. 3 GPIO APB protocol level block diagram

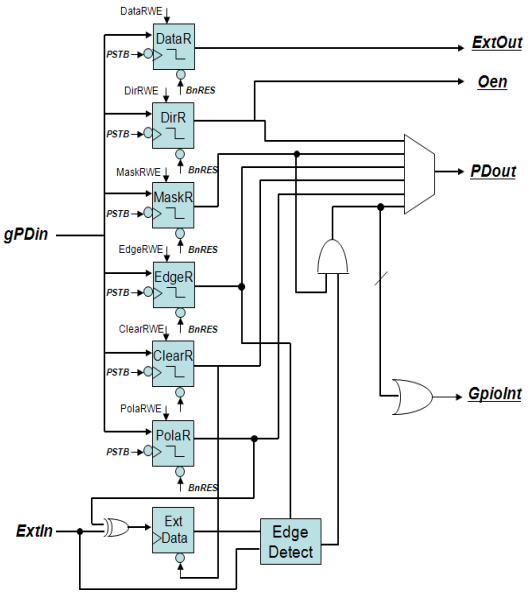

그림 4. GPIOsub8 내부 블록 회로도
Fig. 4 GPIOsub8 internal circuit diagram

표 1. GPIO 레지스터 요약
Table. 1 GPIO register summary

Address	Name	Width	R/W	Default
0x8002.3000	PADataR	8	R/W	0x00
0x8002.3004	PADataDirR	8	R/W	0xff
0x8002.3008	PAIntMask	8	R/W	0x00
0x8002.300c	PAIntStatus	8	R	0x00
0x8002.3010	PAIntEdge	8	R/W	0x00
0x8002.3014	PAIntClear	8	W	0x00
0x8002.3018	PAIntPolarity	8	R/W	0x00
0x8002.301c	GPIOAMUXCtl	7	R/W	0x00
0x8002.3020	PBDataR	8	R/W	0x00
0x8002.3024	PBDataDirR	8	R/W	0xff
0x8002.3028	PBIntMask	8	R/W	0x00
0x8002.302c	PBIntStatus	8	R	0x00
0x8002.3030	PBIntEdge	8	R/W	0x00
0x8002.3034	PBIntClear	8	W	0x00
0x8002.3038	PBIntPolarity	8	R/W	0x00

V. 구현 및 시뮬레이션

설계한 GPIO는 Verilog HDL로 구현하였으며, 자동 합성은 Altera의 QuartusII 통합 환경을 사용하고 시뮬레이션은 ModelSim을 사용하였다..

GPIO의 특성상, 검증은 양방향 PD 버스와 외부입력 포트 PORTIn를 통하여 데이터를 쓰고, 읽어보는 방식으로 행하였다. GPIO의 자원 사용량은 표 2에 나타내었고, 포트 A에 대한 시뮬레이션 파형을 그림 5에 보였다. 검증은 3단계로 진행하였는데, 첫 번째로 PWRITE가 low인 파형의 왼쪽 부분은 내부 레지스터의 초기값을 읽는 과정이다. 가운데 부분은 PWRITE 신호를 주고 PA를 바꿔가면서 DataDir, IntMask, IntEdge, IntClear, IntPolarity 레지스터에 값을 넣고, PWRITE 신호를 해제하여 확인하는 과정이다. PD 버스를 통해서 올바르게 저장된 데이터가 출력되는 것을 확인할 수 있다. 이 때, Polarity를 바꿔주면 원래 active high에서 동작해야 하는 인터럽트가 active low로 바뀌면서 발생하게 된다. IRQ를 클리어 해준 이후 오른쪽 부분에서는 외부 패드 입력인 PORTAIn으로 데이터를 전송하여 PD 버스를 통해 내부 마이크로프로세서로 전달되는 과정을 볼 수 있으며, 외부 입력으로 인한 인터럽트가 또 발생한다. 이와 같이, 포트A에 데이터를 AMBA APB 버스 프로토콜과 외부 핀을 함께 사용하여 저장되고 읽혀지는 과정을 확인할 수 있다. 설계한 GPIO 모듈은 타겟 블루투스 임베디드 시스템의 동작주파수인 25Mhz에서 충분히 정상적으로 동작하였다. 타겟 FPGA는 CycloneII를 사용하였고, 총 256개의 LUT를 사용하였으며 162개의 레지스터가 자동 합성되어 사용되었다.

표 2. 자원 사용량
Table. 2 Resource utilization

사용 자원	사용량
Combinational ALUTs	256
Dedicated logic registers	162

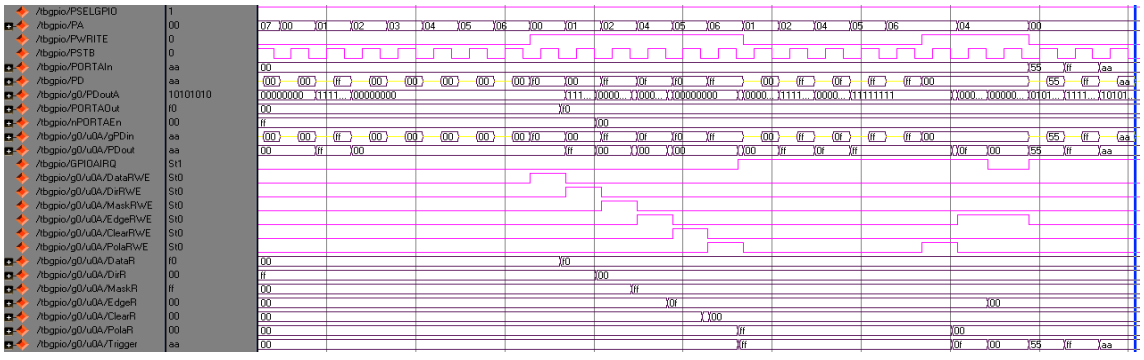

그림 5. GPIO RTL 시뮬레이션 파형
Fig. 5 RTL simulation waveform of GPIO

VI. 결 론

GPIO란 다용도 목적으로 사용되는 입출력핀이라고 말할 수 있는데, GPIO를 이용해서 키입력을 받을수도 있고 특정 Device를 제어할 수 있고, 소프트웨어에서 어떤 특정한 상황이 일어났을때 어떤 특정장치가 동작하게 할 수 있는 등 임베디드 시스템에 필수적으로 사용되는 다용도 입출력으로 사용할 수 있는 인터럽트 모듈이다.

본 논문에서는 블루투스 임베디드 시스템을 위한 다양한 모드를 가지는 GPIO를 설계하고, Altera의 CycloneII의 타이밍 검증 요소를 가지고 시뮬레이션 하여 결과를 확인하였다. 설계한 GPIO는 기본적인 방향성, 트리거 방식 (에지 혹은 레벨), 극성 선택 등 블루투스 시스템에서 발생 가능한 모든 경우의 기능을 유추하여 탑재하여 다양한 응용에 사용할 수 있을 것으로 기대된다. 추후 follow-up 과제로는 이렇게 설계한 GPIO를 실제로 ARM 프로세서와 연결시켜 PC 상에서 사용자 프로그램을 통하여 GPIO를 동작시킬 수 있는 임베디드 시스템을 구현할 수 있을 것으로 기대된다.

참고문헌

[1] http://www.bluetooth.org

[2] Bluetooth Specification Version 4.0 [vol.2] *Radio Specification,* Core System Package [BR/EDR Controller volume] Part A.

[3] http://en.wikipedia.org/wiki/Bluetooth_Special_Interest_Group

[4] Specification Volume 1, *Specification of the Bluetooth System, Architecture and Terminology Overview,* Covered Core Package version: 4.0.

[5] Hynix Semiconductor, "Bluwave HBN3031 - Bluetooth Single Chip"

[6] Das, A. et al., "Adaptive link-level error recovery mechanisms in Bluetooth," Personal Wireless Communications, 2000 IEEE International Conference on , 17-20 Dec. 2000 pp. :85~89

[7] Cheol-Hee Park et. al., "Design and implementation of error control algorithms for Bluetooth system: open-loop and closed-loop algorithms," Consumer Electronics, 2000. ICCE. 2000 Digest of Technical Papers. International Conference on , 13-15 June 2000, pp. 302~303

[8] Y.Taur and T.H.Ning, Fundamentals of Modern VLSI Devices, Cambrigde, U.K.: Cambrigde Univ. Press, 1998.

저자소개

문상국 (Sangook Moon)

1995 연세대학교 전자공학 학사
1997 연세대학교 전자공학 석사
2002 연세대학교 전자공학 박사
2002~2004 하이닉스반도체 선임연구원
2004~현재 목원대학교 전자공학과 부교수
※관심분야 : 정보보호 VLSI 설계, Data encryption, Embedded SoC Systems