EPC RFID 프로토콜 제너레이션 2 클래스 1 태그 디지털 코덱 설계

정회원 이 용 주*, 준회원 조 정 현*, 김 형 규*, 김 상 훈*, 정회원 이 용 석**

Design of Digital Codec for EPC RFID Protocols Generation 2 Class 1 Codec

Yong-joo Lee* ***Reguler Member***, Jung-hyeon Jo*, Hyung-kyu Kim*,
Sang Hoon Kim* ***Associate Member***, Yong-Surk Lee** ***Regular Member***

요 약

본 논문에서는 RFID 표준 중의 하나인 EPC 글로벌 제너레이션 2 클래스 1(EPC global generation 2 class 1) 태그의 설계에 대하여 논하였다. RFID 표준에 관한 연구나 충돌 방지(anti-collision) 알고리즘에 관한 연구는 많이 진행이 되었지만 태그 디지털 코덱 아키텍처 하드웨어의 구체적인 설계에 관한 논문은 아직 없는 실정이기 때문에 본 논문에서 연구하게 되었다. 본 논문의 목적은 RFID 태그 블록의 구성 및 기능 설계에 관한 연구를 함으로써 대략적인 전력 소모, 하드웨어 크기 등에 대한 방향을 제시하고 있다. 스탠더드 셀 라이브러리 합성방식을 사용하여 합성한 결과 설계된 디지털 코덱의 크기는 111640.328125개(인버터 개수)였고 소모 전력은 동적 소모 전력을 기준으로 10.3575uW로 추정되었다. 풀커스텀(full-custom)방식을 사용할 경우, 더욱 개선된 효과를 발휘할 것으로 보인다.

Key Words : RFID, Low power Tag, Digital codec, air protocol, EPC generation 2

ABSTRACT

In this paper, we designed a digital codec of an RFID tag for EPC global generation 2 class 1. There are a large number of studies on RFID standard and anti-collision algorithm but few studies on the design of digital parts of the RFID tag itself. For this reason, we studied and designed the digital codec hardware for EPC global generation 2 class 1 tag. The purpose of this paper is not to improve former studies but to present the hardware architecture, an estimation of hardware size and power consumption of digital part of the RFID tag. Results are synthesized using Synopsys with a 0.35um standard cell library. The hardware size is estimated to be 111640 equivalent inverters and dynamic power is estimated to be 10.4uW. It can be improved through full-custom design, but we designed using a standard cell library because it is faster and more efficient in the verification and the estimation of the design.

Ⅰ. 서 론

현재 국내외적으로 표준화가 진행되고 있고 정부의 주도하에 점차적으로 제품화를 위한 연구가 되어가고 있는 RFID(Radio Frequency IDentification) 분야의 태그의 디지털 코덱을 설계하였다. 13.56MHz에 대응하는 제품들은 거의 상용화에 이르렀지만 UHF대역에 대응하는 ISO 18000-6이나 EPC global

* 연세대학교 전기전자공학과 프로세서연구실 (leemann@dubiki.yonsei.ac.kr)
논문번호 : KICS2005-10-399, 접수일자 : 2005년 10월 4일, 최종논문접수일자 : 2006년 3월 8일

에 대해서는 아직까지 표준화가 완성되지 못하였다. 때문에 본 논문에서는 국제 표준 중에 UHF 주파수 대역에서 가장 많이 쓰이는 ISO 18000-6과 EPC global 표준 중에서 EPC global class 1을 지원하는 태그의 디지털 코덱의 설계에 대한 내용을 담고 있다. 아직 확정된 표준화가 이루어지지는 않았기 때문에 국내외적으로 UHF 대역의 표준을 지원하는 RFID 태그의 디지털 파트에 대한 제품은 물론이고 논문 또한 거의 없기 때문에 본 논문에서는 앞으로 설계되어 제품화될 RFID 태그의 면적 및 소모 전력의 예상 자료로서 사용될 수 있을 것이다.

Ⅱ. 기존 연구 동향 및 표준 연구

2.1 현재까지의 연구동향

기존의 연구의 내용을 살펴보면 RFID 자체의 설계보다는 아직 표준이 확정되지 않았기 때문에 태그간의 간섭을 최소화하려는 충돌방지에 관한 연구[1]와 정책 및 표준안에 대한 연구[2] 그리고 RFID 환경을 이용한 유비쿼터스 등의 다양한 애플리케이션에 대한 연구[3]가 많이 진행되고 있다. 참고 문헌에는 그 중에 하나씩만 실었다.

2.2 기존 연구의 부족한 점

기존에 연구된 RFID는 그 표준안과 어플리케이션 그리고 서비스에 국한되어 있다. 하지만 태그 내부의 UHF 대역의 실제적인 디지털 코덱의 하드웨어 설계 내용에 관해서는 아직까지 연구가 기업에 국한되어 논문으로 나온 것이 국내외를 통틀어 거의 없는 실정이다. 전체 태그의 설계의 중요부분이 RF단의 아날로그 단이고 실제적으로 전력의 많은 부분을 소비하지만 RFID 규격의 내용을 지원하고 리더와 통신을 하고 정보를 주고받는 부분인 디지털 코덱의 하드웨어적인 설계 또한 중요한 부분을 차지한다. 하지만 이 부분에 대한 연구가 미흡하기 때문에 실제로 태그칩을 설계할 때, 아날로그 부분과 메모리 부분을 제외한 실제 코덱 부분의 구현에 대한 구체적인 알고리즘이나 면적 및 소모 전력에 대한 연구가 거의 없다.

2.3 EPC global 표준

2.3.1 EPC 표준의 종류 및 특징

EPC(Electronic Product Code)는 MIT의 Auto-ID Center에서 제정한 RFID 표준으로서 class 0에서 태그의 사용 용도는 수퍼마켓에서 상품의 계산이나 진열 상품의 스캔 또는 일반적인 짐 파렛(pallet)의 식별 또는 창고의 문을 통해 들어오는 상품 식별, 그리고 창고의 관리에 사용될 수 있다.[4] Class 0의 특징은 일반적으로 보안성이 요구되지 않는 매우 간단한 응용에 사용하기위한 표준으로 제정되었다. 다음의 표 1은 EPC global class 0에서 정의한 EPC의 구성 및 크기에 대한 자료이다.[4]

표 1. EPC 구성

EPC Type	Header Size	First Bits	Domain Manager	Object class	Serial Number	Total
64 bit type I	2	01	21	17	24	64
64 bit type II	2	10	15	13	34	64
64 bit type III	2	11	26	13	23	64
96 bit and more	8	00	28	24	36	96

EPC의 구성은 버전 넘버, 도메인 매니저 넘버, 오브젝트 클래스, 시리얼 넘버의 네 가지로 구분된다. 버전 넘버는 여러 가지 EPC를 구분하기 위한 숫자이고, 매니저 넘버는 생산자 구별 숫자이고, 오브젝트 클래스는 생산물의 인식 번호이고, 시리얼 넘버는 일련번호이다.[4]

본 논문에서 표준으로 설정하여 설계한 EPC generation 2의 class 1의 태그의 특징은 class 0에 링크 보안성(link security) 기능과 태그를 무효화시키는 킬(kill)기능을 추가하였고, EPC의 오브젝트 식별자(Object IDentifier)가 상대적으로 적은 수만을 할당할 수 있기 때문에 이 부분만을 확장하여 갖고 있다.[5] 다음의 표 2는 각 클래스 태그들의 특징을 요약한 것이다.[5]

표 2. 클래스의 종류 및 특징

클래스	종류	특징	추가사항
클래스1	Identity Tag	Passive	• Object Identifier(OID) • Link Security • Kill feature
클래스2	Higher Functionality Tag	Passive	• Tag Identifier(TID) • Optional read/write user memory
클래스3	Semi-Passive Tag	Semi-Passive	• On-tag energy source to power logic functionality
클래스4	Active Tag	Active	• Tag-to-tag communications • Active communications • Ad-hoc networking capabilities

상위의 클래스들은 모두 하위 클래스의 기능을 포함하게 되어있다.

2.3.2 EPC global generation 2 class 1

EPC global generation 2 class 1에서 정의한 태그의 동작은 다음은 그림 1과 같은 상태 다이어그램으로 표현할 수 있다.

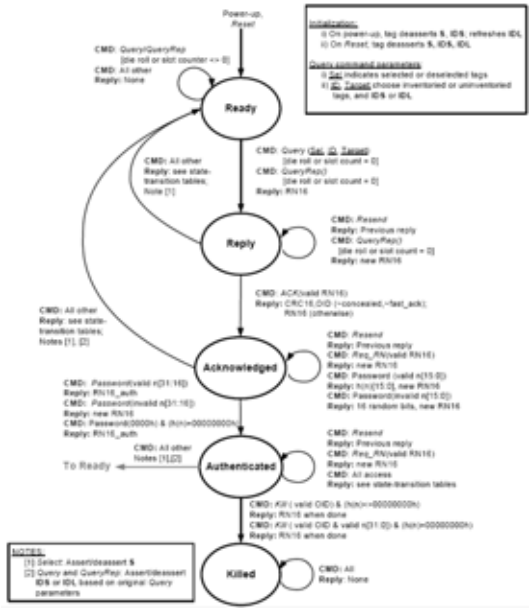


그림 1. 태그 상태(state) 다이어그램

표 3. 명령어 종류 및 설명

명령어	코드	길이(bit)	필수여부
Query	00	19	Yes
QueryRep	01	8	Yes
ACK	1000	27	Yes
NAK	1001	10	Yes
Select	1010	21 or >39	Yes
Resend	1011	14	Yes
Req_RN	11000000	38	Yes
ResetR	11000001	22	Yes
Read	11000010	>55	No
Write	11000011	>63	No
Lock	11000100	40 or >47	Yes
Kill	11000101	54	Yes
Concea	11000110	39	No
Password	11000111	39	Yes
WritePassword	11001000	55	No
Base commands reserved for future use	11001001 11011111	-	-
Extended commands reserved for future use	11100000 00000000 .... 11101111 11111111	-	-

태그는 총 5가지의 상태를 갖고 있다. 처음의 전력이 들어왔을 때, 리셋(Reset) 명령을 받았을 때 또는 잘못된 명령이나 잘못된 값의 응답을 받았을 때, 준비(Ready) 상태가 된다. 쿼리(Query) 명령을 받았을 때, 카운터나 랜덤 숫자가 0이 되어서 태그가 응답하게 된다면, 태그는 응답(Reply) 상태로 넘어가게 된다. 이 상태에서 ACK명령을 받게 되면 태그는 승인(Acknowledgement) 상태로 넘어가게 되고 일반적인 태그의 인식은 이 상태에서 끝나게 된다. 하지만 메모리 액세스를 위해서는 이 상태에서 패스워드 명령어를 이용하여 인증(Authenticated) 상태로 넘어가야 한다. 인증상태에서만 태그를 무효화 시킬 수 있는 킬(Killed) 상태로 갈 수 있다.

2.3.2 명령어(Command)

EPC generation 2 class 1 태그에서 지원하는 명령어는 크게 2 가지로 구분된다. 꼭 구현해야 하는 필수 명령(required command)과 선택 명령(optional command)이다. 본 논문에서는 필수 명령과 선택 명령 모두를 지원하도록 설계된 내용을 담고 있다. 다음의 표 3은 각 명령어들을 정리한 것이다.

Ⅲ. 제안된 태그 설계

서론에서도 밝힌 듯이 태그의 물리적인 측면이나 충돌방지 그리고 정책에 관하여서는 많은 연구가 진행 중이고 많은 논문이 발표되었지만 실제의 태그 내부의 디지털 코덱의 구현에 관한 논문은 아직 없는 현실이다. 본 챕터에서는 표준 상에 있는 커맨드의 실제 구현 방법에 대하여 연구한 내용을 기술하였다.

3.1 태그의 블록 구성

본 논문에서 설계된 태그의 디지털 코덱의 블록 구성은 다음의 그림 2과 같다. 그림에서 보면 크게 7가지의 블록으로 나뉜다.

3.1.1 프리앰블 검출기(Preamble detector)

3장의 그림 2의 프리앰블 신호에서 위반신호(violation)은 실제 통신에서 쓰일 수 없기 때문에 앞의

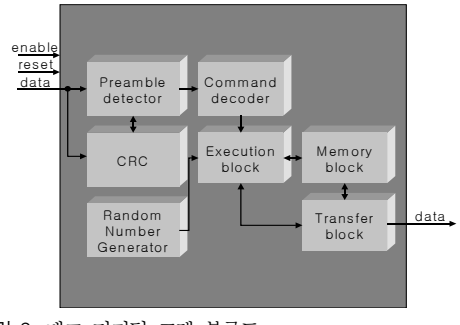


그림 2. 태그 디지털 코덱 블록도

연속된 4개의 0과 뒤의 0을 사이에 위반신호가 들어오면 이 신호는 리더에서 보내는 명령의 시작을 나타내게 되고 이 신호를 수신한 태그는 다음의 값부터 CRC(Cyclic Redundancy Check)블록을 통해 계산을 하면서 명령어 해석(command decoder)블록으로 값을 넘겨주게 된다.

3.1.2 CRC

CRC8을 지원하도록 설계되어 있다. CRC8의 정의는 다음의 표 4와 같다.

표 4. CRC8 정의

CRC8 Definition				
CRC Type	Length	Polynomial	Preset	Residue
—	8 bits	$x^8+x^4+x^3+x^2+1$	FF_h	none

3.1.3 임의수 발생기

임의수 발생기는 표준에서 태그의 식별을 위해 쓰이는 임의수를 발생하는 블록으로서 통신 분야에서 구조가 간단하기 때문에 많이 쓰이는 LFSR(Linear Feedback Shift Register) 방식을 사용하여 구현하였다. 그림 3은 임의수 발생기의 블록도이다. LFSR 방식의 임의수 발생기의 초기에 항상시드(seed)라는 초기값이 있어야 하는데 이 값이 같으면 매 클락마다 같은 패턴의 값이 나오게 된다. 또 하나의 단점은 XOR 게이트에 의해 값이 변화하는데 모든 값이 0이 되면 주기(cycle)가 지나도 값이 변하지 않는다. 이를 해결하기 위해서 복잡한 방식을 지양하고 간단히 메모리의 리저브드(Reserved) 영역을 이용하여 16-bit의 seed 값을 전력이 공급되는 시점에서 읽음으로서 임의수 발생기를 초기화하도록 하였다. 두 번째의 문제를 해결하기 위해서는 간단히 NOR 게이트를 이용하여 모든 값이 0이 되면 강제로 1의 값을 넣어주도록 하였다.

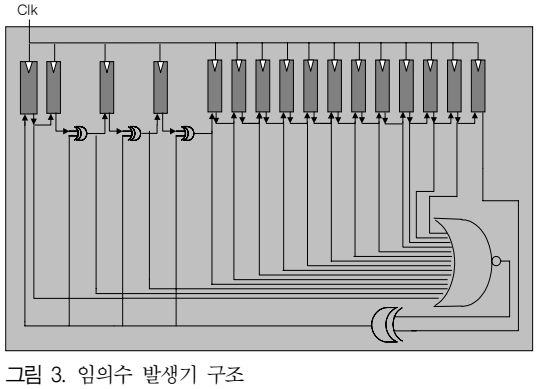


그림 3. 임의수 발생기 구조

3.1.4 명령어 해석기

명령어 해석기는 표준의 명령어 구조가 바이트단위의 데이터 처리에 적합하지 않고 임의의 비트에서 처리하도록 되어있기 때문에 비트 단위로 컨트롤하여 데이터를 처리하도록 되어 있지만 이렇게 설계할 경우, 매 주기 당 처리하는 종류 수가 많아지기 때문에 매우 복잡하게 된다. 따라서 8-bit의 쉬프트 레지스터를 이용하여 바이트 단위로 처리하도록 설계하여 컨트롤이 보다 간단하게 되었다. 비트 단위의 저장을 위해서는 모든 데이터를 쉬프트레지스터로 설계하여야 하지만 비트 시리얼 데이터를 패럴렐 데이터로 변환한 후, 저장하기 때문에 입력단의 쉬프트레지스터 하나를 이용하여 다른 레지스터는 일반적인 레지스터로 설계함으로써 데이터 이동 횟수를 줄임으로써 컨트롤의 복잡성뿐만 아니라 전력 사용면에서도 이득이 있다. 다음의 그림 4는 명령어 해석기 내의 레지스터의 구성을 나타낸다.

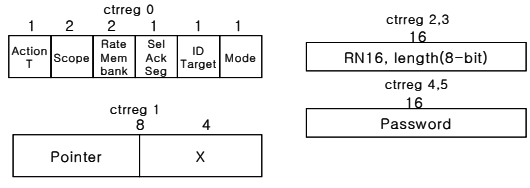


그림 4. 명령어 해석기 내의 레지스터 구성

그림 4를 보면 데이터가 여러 가지가 한 곳으로 입력되게 되어 있는데 같은 곳에 들어가는 데이터는 한 명령어상의 데이터가 한 레지스터에 겹치지 않도록 구성하여 중복 사용하였다. 데이터마다 모두 다른 레지스터를 사용하게 되면 총 레지스터의 수는 64개가 되는데 이를 최적화하여 48개만을 사용하여 25%의 레지스터 수를 줄였다. 하지만 임시로 사용해야 하는 데이터도 공유하여 사용하도록 설계하였기 때문에 실제로는 더 효율이 높다.

3.1.5 실행부

실제 EPC generation 2 class 1의 표준의 코덱을 처리하는 부분이다. 해석기에서 넘어오는 데이터를 이용하여 명령어 종류, 데이터의 내용에 따라 표준에 알맞은 동작을 수행하거나 데이터를 생성하여 리더에게 응답할 데이터를 생성한다. 실행부는 실제의 코덱의 내용을 구현하기 때문에 가장 크고 설계가 까다롭다. 다음의 절들에서는 기본적인 명령어의 동작 및 구현에 대한 설명보다는 태그의 설계에 있어서 까다롭거나 아이디어가 있는 부분에 대해 논

하겠다.

3.1.5.1 선택(Select) 명령어 처리부

선택 명령은 수많은 태그를 모두 인식하기 위해서 각 태그를 선택하는 명령어이다. 이 명령어의 구현이 까다로운 이유는 표준에서는 메모리의 영역을 비트 단위로 리드하여 선택할 수 있도록 규정하고 있기 때문이다. 하지만 논문에서 사용한 메모리의 구조는 바이트(8-bit) 단위(표준에서 정의한 메모리 1 워드는 16-bit)이기 때문에 불려온 데이터를 가공해야 하고 또한 읽어 오는 데이터의 길이가 최대 한 메모리 영역 전부이기 때문에 이 값의 비교 또한 하드웨어적으로 매우 부담이 된다. 다음의 표 5는 선택 명령의 구성을 나타낸다.

표 5. 선택 명령어 구성표

	Preamble	Command	Action	Scope	MemBank	Pointer	Length	Mask	CRC8
# of bits	6	4	1	2	2	EBV	8	Variable	8
Descrip-tion		1010	0: deassert $ 1: assert $	See 10.5.1	00: RFU 01: OID 10: TID 11: User	Starting mask address	Mask length (bits)	Mask value	

표에서 Pointer는 시작 어드레스(address)를 나타낸다. Pointer는 bit 어드레스이고 길이가 가변이기 때문에 다음의 그림 5과 같은 블록으로 설계하였다.

저전력과 저면적의 이유 때문에 표준에서 정의하는 최대 비교 비트 수인 한 메모리 영역 전체의 비교의 경우, 한 메모리 영역이 최대 128 비트라고 가정한다면 128 비트 비교기가 필요하고 또한 이 데이터를 모두 저장할 128개의 레지스터가 필요한데 본 논문에서는 이 구조를 개선하여 8비트 비교기를 사용하여 여러 주기를 통하여 비교하였다.

본 논문의 구조는 8비트의 작은 비교기와 명령어 해석기의 내의 8비트 레지스터를 그대로 이용하였기 때문에 추가적인 레지스터의 필요성도 없다. 단지 마지막 주기까지 결과 1비트를 갖고 있어야 한다는 점과 한번 다른 값을 비교하면 추가적으로 비교하는 블록을 멈추게 하는 추가동작만을 추가하였다. 다음의 표 6은 단일 주기의 128비트 구조와 다중 주기의 8비트 구조의 면적과 소비전력을 비교한 것이다. 합성에 이용한 툴과 라이브러리는 4장에서 설명하기로 한다.

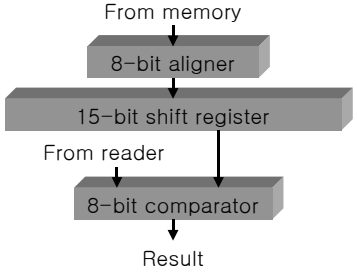


그림 5. 선택 명령 처리부 구조

표 6. 선택 명령 수행 방식 비교

구조	면적	소비전력	비교(면적/전력)
8비트	489.6	1.0968	1.12%/2.27%
128비트	43643.5	48.3078	100%/100%

면적의 1은 인버터의 크기를 나타내고 전력은 동적 전력(dynamic power)이다. 표에서 알 수 있듯이 다중주기 방식의 경우 면적과 전력 면에서 매우 뛰어나다. 하지만 다중 주기의 동작을 해야 하기 때문에 최대 128/8 = 16회의 연산을 해야 한다. 하지만 태그는 전력을 외부 리더의 전파에서 얻기 때문에 순간적인 전력의 양이 문제가 되고 지속적인 전력 소모는 문제가 되지 않는다. 또한 16회 연속 연산을 수행해도 단일 주기 구조에 비해 매우 좋은 특성을 갖는다. 15 비트의 쉬프트 레지스터를 사용한 이유는 8 비트로 정렬된 메모리의 데이터를 리더에서 오는 비트에 맞추어 정렬하기 위해서는 1번 또는 2번의 메모리 읽기가 필요하기 때문이다. 주소가 8의 배수이면 정렬을 할 필요가 없고 최대 7 비트의 정렬이 필요하다. 정렬 과정이 끝나면 주기별로 값을 메모리와 비교하여 같으면 다음을 수행하고 다르면 수행을 멈추고 CRC의 결과를 기다린 후, CRC의 결과에 따라 다음의 동작을 수행하게 된다. 선택 명령은 외부로의 전송이 필요 없고 외부에서 들어오는 데이터와 동기를 맞추어야 하기 때문에 메모리 액세스 명령어 중에서 유일하게 실행부에서 메모리를 컨트롤하는 명령어이다. 선택 명령이외에는 모두 전송부에서 메모리를 컨트롤하도록 되어 있다.

3.1.5.2 재전송 명령

재전송의 경우, 이전의 명령어 또는 이전에 보낸 데이터를 저장할 필요 없이 전송부의 컨트롤 신호만을 기억함으로써 추가적인 페널티 없이 수행할 수 있도록 설계하였다. 다음 장에서 설명할 전송부는 임의수, 특수한 몇 가지 수와 컨트롤 신호만을 보내면 자동으로 데이터를 전송하기 때문에 컨트롤

신호만 유지해 주고 전송 동작 신호만 보내면 재전송이 수행되어 추가적인 하드웨어가 필요 없다.

3.1.6 전송부

전송부의 동작은 실행부에서 전해주는 데이터의 전송뿐만 아니라, 메모리를 직접 액세스하여 데이터를 읽고, 쓸 수 있도록 설계가 되어 있다. 실행부에서 메모리를 액세스하는 경우는 1.5.1에서 설명한 것과 같이 선택 명령뿐이다. 전송부에서 메모리의 액세스 권한을 갖은 이유는 태그에서 리더로의 데이터 전송은 리더에서 태그로의 데이터 전송에 x1, x2, x4 세 가지 모드를 지원해야 하기 때문이다. 실행부는 입력 전송률에 맞추어 동작하는데 입력 전송률과 출력 전송률이 차이가 나기 때문에 메모리 액세스 명령이나 응답이 필요한 명령을 실행부에서 모두 처리한 후, 전송부로 데이터만을 주고 컨트롤할 경우, x2의 경우, 4 주기마다, x4의 경우 2주기마다 전송부로 전송할 데이터를 가변적으로 보내주어야 한다. 이러한 복잡성을 없애기 위해서, 본 논문의 태그에는 전송부 자체가 8 주기마다 메모리를 액세스하여 8비트씩 전송하게 설계하였고, 전송부는 실행부와 달리 1, 2, 4배속의 클록을 이용하여 동작만 하면 간단히 설계할 수 있다. 실행부에서 필요한 처리가 모두 끝난 상태에서 전송부로 컨트롤을 옮기기 때문에 서로 간의 클록차이는 영향을 주지 않는다. 전송부에서 처리되는 모드와 동작은 다음의 표 7과 같다. 읽기 명령과 ACK 명령의 not fast 모드는 모두 메모리의 내용을 읽은 후, 전송하도록 되어 있지만 읽기의 경우에는 16비트 임의수가 추가적으로 전송되기 때문에 비슷한 동작임에도 구분하게 되었다. 임의수 전송은 대부분의 명령어에서 정상적인 처리 후의 응답으로 태그 고유의 임의수를 전송하게 된다.

쓰기 모드는 표준에서는 지원하는 명령어가 아닌 태그의 정보를 전원이 없어져도 유기하기 위해 표준에서 권장하는 동작을 수행한다. 메모리 영역은 보존 영역에 저장이 되고, Lock, Kill, WritePassword, Conceal의 명령어는 그 수행이후 결과에 대한 정보를 메모리에 저장하게 된다. 쓰기 & 임의수 모드는 쓰기 명령어의 수행 시에 동작되는 모드로서, 쓰기 명령을 받은 태그는 쓰기 동작을 성공적으로 수행하였을 때, 임의수를 전송하게 된다. 특정한 수 모드는 lock된 메모리 영역에 쓰기 명령 , 패스워드가 0이고 lock되었을 때 kill명령이나 Conceal 명령, 마지막으로 패스워드가 0이고 lock되었을 때 Write Password 명령이 오면 Null을 지원하지 않는 명령어를 받으면 0xffff를 전송하게 된다. 전송부에는 메모리 컨트롤러와 CRC 생성기, FM0 생성기, sub-carrier 생성기로 구성되어 있다. 위의 모드에 맞는 동작을 수행한 후, CRC를 첨가하여 FM0 또는 sub-carrier 방식으로 응답한다.

표 7. 전송부의 모드와 동작

Mode	동작
읽기	메모리 읽기 명령
not fast ACK	CRC16 & OID 전송
임의수(fast ACK)	임의수 전송(16비트)
쓰기	보존 영역 메모리 쓰기
쓰기 & 임의수	일반 쓰기 명령
특정한 수	특정한 경우 (0x0000 or 0xffff)

3.1.7 메모리

본 논문에서 시뮬레이션을 위해 사용된 메모리의 타이밍도는 다음의 그림 6, 7과 같다.

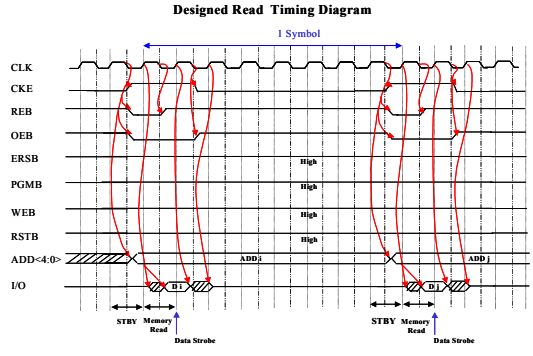


그림 6. 읽기 타이밍도

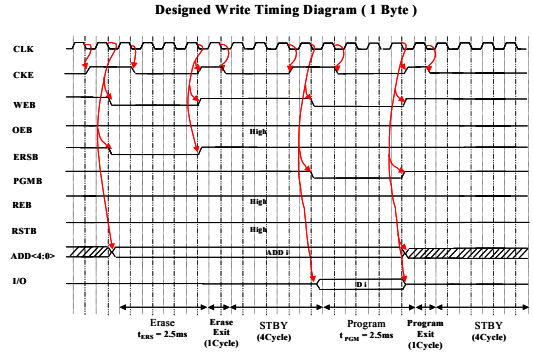


그림 7. 쓰기 타이밍도

그림 6에서 보듯이 읽기는 2 주기가 필요하지만 메모리의 읽기 클록은 320kHz로서 최대 동작 클록인 160kHz보다 2배 빠르기 때문에 실제 1 주기만에 읽기가 가능하여 동작 제어가 매우 용이하다. 쓰기의 경우, 고정적으로 160kHz 클록에서 18 주기가 필요하다. 이렇게 읽기, 쓰기의 타이밍이 다르기 때문에 고정적으로 읽는 방식을 지양하고 핸드세이킹(handshaking) 방식의 구동을 택했다. 메모리 동작 신호가 들어오면 메모리블록은 동작이 완료되면 완료 신호를 줌으로써 동작의 제어를 수행하였다.

Ⅳ. 태그의 설계 결과

4.1 설계 방식 및 테스트

4.1.1 Verilog HDL 설계

태그의 설계는 설계 기간이 짧고 다양한 구조를 쉽게 구현 및 변경할 수 있는 verilog HDL을 이용하여 설계하였다.[6] 설계한 블록의 검증을 위한 테스트 벤치의 내용은 다음 그림 8과 같다. 그림 9는 시뮬레이션 결과의 파형도이다. 신호 data_in의 응답으로 임의수 16-bit를 출력하는 모습이다.

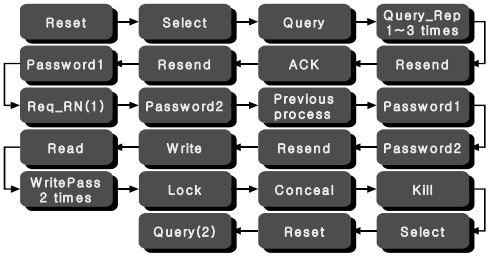


그림 8. 테스트 벤치 구성

그림 9. 테스트 결과 파형도

4.2 합성 환경 및 결과

합성은 sysnopsys의 design analyzer를 이용하여 합성하였다. 스탠더드 셀 라이브러리는 하이닉스 전자의 0.35um공정을 사용하였고 환경은 구동 전압 2.3V 조건은 최악조건으로 하였다. 합성 결과는 다음의 표 8과 같다. 각 블록의 소비 전력은 전체 면적에 비례한다. 총 소비 전력에는 메모리의 전력은 빠져 있다. 메모리 블록은 입출력에 대한 스펙만이 정해져 있었고 실제로 메모리 모델이 설계 중에 있기 때문에 전력 소모의 측정에서 제외하였다. 설계 시에는 모두 레지스터를 이용하여 메모리 블록을 시뮬레이션하였다. 그렇기 때문에 디지털 코덱 부분에서는 메모리의 제어부에 대한 면적과 전력을 계산하였다. 또한 계산된 전력은 동적 전력만을 포함하였다. 파워 컴파일러를 이용하여 그림 8의 테스트 벤치 입력에 대한 동적 전력을 시뮬레이션하였고 그 결과를 표 8에 정리하였다.

표 8. 합성 결과

	면 적 (# of inverters)	소비 전력 (uW)
프리앰블 검출기	2188.799988	
CRC8	1497.599976	
명령어 해석기	24560.636719	
실행기	5666.882812	
전송부	18357.119141	
기타 블록	53494.089537	
합 계	111640.328125	10.3575

Ⅴ. 결 론

앞서 이야기 한 바와 같이 RFID 분야에서 기존에 많이 연구된 표준과 충돌 방지에 관한 연구를 통하여 실제로 하드웨어적으로 설계된 연구가 거의 없기 때문에 본 논문의 결론은 기존의 설계된 하드웨어의 단점을 개선하고 보완하는 것보다 전력 소모량에서 기존의 연구 논문을 인용하여 다른 관점에서 비교하였다. 논문에서 설계한 RFID용 EPC global generation 2 class 1을 지원하는 태그의 설계에 관한 아키텍처적인 관점에 중심을 두어서 설계를 하였다. 스탠더드 셀 합성방식의 설계이기 때문에 전력적인 면에서는 다소 부정확하고 실제에 적용하기 힘든 면이 있다. Feng Jhou의 논문에 의하면 UHF(900MHz)대역의 태그에 최대 인가될 수 있는 전력이 100uW이하라는 글이 있다. 본 논문에서 설계된 부분에서 전력을 추정한 부분은 디지털 코덱만이고 스탠더드 셀 방식으로 설계되어 합성한 부분의 전력만을 측정하였고 메모리와 RF단의 전력 등은 고려되지 않았지만, 전력적인 부분을 최적화하여 설계한다면 표 7의 결과와 같이 100uW이하의 충분히 저전력 구조로 설계가 가능할 것으로 보인다.[7]

참 고 문 헌

[1] Feng Zhou, Chunhong Chen, Dawei Jin,

Chenling Huang, Hao Ming, “Evaluating and Optimizing Power Consumption of Anti-Collision Protocols for Application of RFID systems.”, *Proceedings of the 2004 International Symposium*, pp 357-362, 2004

[2] 신상철, 김유정, 송석현, “RFID/USN 국제 표준화 대응 전략 및 보급 활성화 방안”, *한국통신학회지 제21권 6호*, pp 22-38, 2004

[3] 전성훈, 전호인, “IEEE 802.15.4 and ZigBee Protocol : 유비쿼터스 센서 네트워크를 위한 Active RFID 기술”, *한국통신학회지 제21권 6호*, pp 67-88, 2004

[4] “Draft protocol specification for a 900 MHz Class 0 Radio Frequency Identification Tag”, *MIT Auto-ID Center*, 2003

[5] “EPC Radio-Frequency Identity Protocols Generation 2 Identity Tag(Class 1): Protocol for Communications at 860 MHz-960 MHz”, *EPCglobal*, 2003

[6] Donald E. Thomas, Philip Moorby, “The Verilog Hardware Description Language”, KLUWER ACADEMIC PUBLISHERS

[7] Feng Jhou, | Dawei Jin, Chenling Huang, Min Hao “Optimize the Power Consumption of Passive Electronic Tags for Anti-collision Schemes”, ASIC, 2003. Proceedings. 5th International Conference on Volume 2, 21-24 Oct. 2003 Page(s):1213-1217 Vol.2

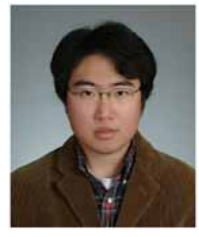
조 정 현 (Jung-hyun Jo) 준회원

2004년 8월 연세대학교 기계전자공학부 졸업

2004년 9월~현재 연세대학교 전기전자공학과 석사과정

<관심분야> 전자공학, 마이크로프로세서, ASIC

김 형 규 (Hyung-kyu Kim) 준회원



2004년 2월 연세대학교 전기전자공학부 졸업

2004년 8월~현재 연세대학교 전기전자공학과 석사과정

<관심분야> 전자공학, 마이크로프로세서, ASIC

김 상 훈 (Sang-hoon Kim) 준회원

2005년 2월 서울산업대학교 전자공학과 학사취득

2005년 3월~현재 연세대학교 전기전자공학과 석사과정

<관심분야> 전자공학, 마이크로프로세서, ASIC

이 용 주 (Yong-joo Lee) 정회원

1999년 8월 연세대학교 전자공학과 졸업

2001년 8월 연세대학교 전기전자공학과 석사학위 취득

2001년 9월~현재 연세대학교 박사과정

<관심분야> 전자공학, 마이크로프로세서, ASIC

이 용 석 (Yong-surk Lee) 정회원

연세대학교 전기공학과 졸업

Intel Corp., Santa Clara,California, USA , 펜티엄 마이크로프로세서 설계

1993년현재 연세대학교 전기전자공학과 교수로 재직중

<관심분야> 전자공학, 마이크로프로세서, ASIC