

Applications & Practice

THE UNWINDING OF A PROTOCOL

Swapna Dontharaju, Shenchih Tung, Alex K. Jones, Leo Mats, Justin Panuski, James T. Cain, and Marlin H. Mickle

ABSTRACT

The ISO 18000 Part 6C UHF standard is becoming a widely accepted standard in RFID applications in supply chain management and is driving development of passive tags. The communication primitives of ISO 18000 Part 6C are significantly different and more complex than ISO 18000 Part 7. The complexity of the Part 6C standard makes the design of these tags extremely time consuming and challenging for reducing power consumption and silicon area. This article examines various features of the ISO Part 6C standard and compares it to the ISO Part 7 standard for active tags for the purpose of evaluating generic interrogator/tag protocol complexity. For a 0.16 mm ASIC implementation, the Query command from ISO 18000 Part 6C is more complex than 10 primitives of the simpler ISO 18000 Part 7 standard.

INTRODUCTION

Radio frequency identification (RFID) tags and other small autonomous devices are typically powered by a continuous wave (CW) of RF energy (passive) or make use of a small battery (active). In either case the object of the design and the communication protocol between an interrogator (reader) and the device (tag or sensor) is to reduce energy consumption. The communication is a frame of digital bits encoded in some form including a preamble, command/data, and checksum/cyclic redundancy code (CRC). The device is typically implemented with a silicon chip where the design objectives are to reduce cost, power, and voltage. One of the difficulties in obtaining these objectives is complexity.

In terms of design, there are several alternatives to implement the set of commands and responses to support communication between interrogator and tag. Some of these protocols are the ANSI 256 [1] and ISO 18000 Part 7 active standards [2], the proposed ISO 18185 Part 1 RFID container seal standard [3], and the EPCglobal Class 1 Gen-2 (C1 G2) specification [4] recently standardized as the ISO 18000 Part 6C passive standard [5]. These standards are for ultra high frequency (UHF) tags.

The communication frames between interrogator and tag are frequently referred to as command lines or *primitives*. In this article we use the term primitive. The primitive, in addition to a synchronization preamble and CRC code, contains fields that are decoded by the receivers. There are trade-offs in the complexity of the fields of individual commands and the number of commands. Typically, one field indicates the type of command, much like an *opcode* for a processor instruction, with a number of sub (or additional) fields. There are trade-offs between the size of the command field (e.g., number of commands in the system) vs. the complexity of the

device. Often, as is the case for ISO 18000-6C, the complexity is related to the number of states in the system. For example, a bigger command field with many different system commands reduces the complexity of individual commands. In addition, there are fewer subfields per command, and the response logic for any given command may be simplified. For example, an increased number of commands reduces the complexity of the states of the device.

The complexity of the logic on the chip has been examined through a primitive (command line) to VHDL compiler that has been developed at the University of Pittsburgh [6, 7]. The object of this compiler is to go directly from a set of commands (both interrogator and device) to VHDL to rapidly prototype any interrogator or device, and to evaluate both the power required for the device implemented in silicon and the robustness of the protocol.

Recognizing the need for over-the-air security, there is a general nature to rely on the state of the target device to facilitate security encoding. Reverse engineering a device with a significant amount of state information is much more difficult than the same task on a device with little or no state information. This is in many ways analogous to the verification problem for logic synthesis of finite state machines. While this is not a particular design problem for battery-powered active devices, passive devices can be problematic for temporary storage of keys and intermediate states as there is no dedicated power source.

These passive protocols work correctly during prototyping using field programmable gate array (FPGA) development boards with uninterrupted power sources. However, the incorporation of a temporary bit or bits or non-volatile storage in silicon is problematic for technologies that are cost driven at the foundry. ISO 18000 Part 6C is an example of a protocol that relies on intermediate storage to retain the state of the system.

We explore the impact of the protocol specification on the implementation and, in particular, the design complexities and their impact on power consumption and expected fabrication cost. As a case study we examine the ISO 18000 Part 7 active standard in comparison to the ISO 18000 Part 6C passive standard. As an analogy, the ISO 18000 Part 7 standard is analogous to a reduced instruction set computer (RISC) style processor, while the ISO 18000 Part 6C protocol is analogous to a complex instruction set computer (CISC) style processor. Using our RFID compiler, we implemented these different protocols and show the difference in complexity, and the impact on silicon power and area.

The remainder of this article is organized as follows. We describe some background and relevant previous work in the development of RFID systems. Introductions to the ISO 18000 Part 7 and Part 6C standards are presented, respective-

ly. We describe how the RFID compiler can be used to implement ISO 18000 Part 7 and Part 6C using a custom hardware description. We present results of implementing ISO 18000 Part 7 and Part 6C using our design flow. Finally, conclusions and future directions are presented.

BACKGROUND AND RELATED WORK

There has been a recent explosion of interest in RFID. Research and development in RFID has led to its deployment in numerous applications, including supply chain management, automatic toll collection, retail stores, location sensing, libraries, healthcare, airports, animal tracking, and building access control.

Some of the open issues in the RFID domain are the existence of multiple standards, handling of stored data, tag orientation, reader collision [8], range, cost, and security concerns [9]. For active tags, maximizing battery life is an important concern. To improve RFID security, research groups have proposed novel solutions such as the blocker tags [10] and implemented authentication using encryption [11]. Recent advancements in tag hardware contribute to improved range and power consumption [6], improved tag antenna design [12], packaging, and tag orientation.

The features of the ISO 18000 Part 6C (previously “Gen 2”) protocol and their implications for asset management have been previously studied [13]. The necessity for a uniform organization of user memory and the related omissions in the EPC protocols are addressed in [14].

ISO 18000 PART 7

The ISO 18000 Part 7 standard [2] is an international standard that defines the air interface for RFID devices used in item management applications. The standard defines the forward and return link parameters for an active RFID air interface at 433 MHz and the communications protocol used.

PHYSICAL LAYER

The interrogator transmits information to the tag by modulating an RF signal in the $433.92 \text{ MHz} \pm 20 \text{ ppm}$ frequency range. Modulation is effected with frequency shift keying (FSK) with deviation of $\pm 35 \text{ kHz}$ and the modulation rate is 27.7 kHz. A 30 kHz wake up signal is transmitted by the interrogator for a minimum of 2.5 seconds to wake up all tags within communication range. The interrogator always initiates communication with the tag and subsequently listens for a response.

DESCRIPTION OF INTERROGATOR COMMANDS

Table 1 shows the ISO 18000 Part 7 Commands and the command codes used to identify them. Custom and proprietary commands are supported by the protocol and will use command codes in the range A0–FF.

Interrogator to Tag Command Format — Figure 1 shows the interrogator to tag command format. Each command contains a *command code* to signal the tag what type of command is being issued. Additionally, the command always contains a CRC to ensure the command packet is properly formed. The remainder of the packet contains particular fields appropriate to the command. For example, the command type field indicates the presence of *Tag ID* and *Owner ID* fields. Broadcast commands do not contain a Tag ID, while point-to-point commands contain a specific Tag ID of the target tag. The Owner ID field, which is programmed in the tag’s memory, allows the segregation of different groups of tags within the whole population.

Response Format — The tag response has two different for-

Command code (R/W)	Command name	Command type
0x10/NA	Collection	Broadcast
0x11/NA	Collection with Data	Broadcast
0x14/NA	Collection with User ID	Broadcast
NA/0x90	Sleep	Point to point
0x01/NA	Status	Point to point
0x07/0x87	User ID Length	Point to point
0x13/0x93	User ID	Point to point
0x09/0x89	Owner ID	Point to point
0x0C/NA	Firmware Revision	Point to point
0x0E/NA	Model Number	Point to point
0x60/0xE0	Read/Write Memory	Point to point
0xNA/0x95	Set Password	Point to point
0x17/0x97	Set Password Protect	Point to point
NA/0x99	Unlock	Point to point

■ **Table 1.** ISO Part 7 command codes.

mats depending on the type of response. Figure 1 shows the response format for a broadcast command and a point-to-point command. Broadcast commands are used to collect information from a selected group of tags (e.g., *Collection* command). The tag responses also include a tag status field, which consists of nested fields such as acknowledge, tag type, and battery.

Security Features — To prevent malicious access to the tags, a password style security mechanism is provided in the standard by the *Set Password*, *Set Password Protect*, and *Unlock* commands. The Set Password command sets the tag’s password and requires that the tag be unlocked before it can be accessed. The Set Password Protect command sets or clears the password protection condition of tag. All point-to-point commands can be password protected using this command. The Unlock command unlocks the access to the tag. The Owner ID and *User ID* provide additional layers of privacy to the tag. User ID is an optional field, up to 16 bytes long, which allows the user to program his own user assigned asset identifier. The *Command Type* field in the interrogator command indicates if this field is used in the communication. The User ID command defines or determines the User ID, and the *User ID Length* command defines or determines its length.

ISO 18000 PART 6C

ISO 18000 Part 6C [5] is a recent amendment to ISO 18000 Part 6 that describes the RFID air interface for devices operating at 915 MHz and the communications protocols used. Part 6C extends the existing Part 6 standard, which previously

Interrogator to tag command format							
Command prefix	Command type	Owner Id	Tag Id	Interrogator Id	Command code	Parameters	CRC
1 byte	1 byte	3 bytes	4 bytes	2 bytes	1 byte	N bytes	2 bytes

Broadcast response format							
Tag status	Message length	Interrogator Id	Tag Id	Owner Id	User Id	Data	CRC
2 bytes	1 byte	2 bytes	4 bytes	3 bytes	0-16 bytes	0-N bytes	2 bytes

Point to point response format						
Tag status	Message length	Interrogator Id	Tag Id	Command code	Parameters	CRC
2 bytes	1 byte	2 bytes	4 bytes	1 byte	N bytes	2 bytes

■ **Figure 1.** ISO Part 7 command and response formats.

contained type A and B devices with a type C modeled after the EPCGlobal C1 G2 specification.

PHYSICAL LAYER

An interrogator transmits information to a tag by modulating an RF signal in the 860–960 MHz frequency range. These devices use one of three modulation techniques: double-sideband amplitude shift keying (DSB-ASK), single-sideband ASK (SSB-ASK), or phase reversal ASK (PR-ASK) in addition to using a pulse interval encoding format. The tag receives its operating energy from this modulated RF waveform. To receive information from a tag, the interrogator transmits an unmodulated RF signal to the tag and listens for a backscattered reply. The tag responds by modulating the amplitude or phase of the RF signal. The encoding format can either be FM0 or Miller modulated subcarrier. Interrogators and tags use a half duplex communication link.

DESCRIPTION OF INTERROGATOR COMMANDS

An interrogator manages tag populations using three basic operations, select, inventory, and access. The selection concept is similar to broadcasts using the Owner ID from ISO 18000 Part 7. The *Select* command is applied successively to pick a particular tag population based on user-specific criteria, enabling union, intersection, and negation based tag partitioning. An interrogator begins an “inventory round” by transmitting a *Query* command in one of four sessions. One or more tags may reply to this. The interrogator then detects a single tag reply and requests more information from the tag. The access concept is similar to point-to-point communications in ISO 18000 Part 7. The access operation allows the issuing of commands that read from or write to a tag once the tag is uniquely identified.

Table 2 shows the 18000 Part 6C commands and their command codes. Custom and proprietary commands are supported by the protocol with command codes in the range E000–E1FF.

TAG STATES AND SLOT COUNTER

ISO 18000 Part 6C tags implement features such as accessing and killing passwords, checking the electronic product code (EPC), CRC checking, manipulating the slot counter, and pseudo-random number generation. The states and keys of the target device are used to facilitate tag singulation, collision arbitration, security encoding, and so on.

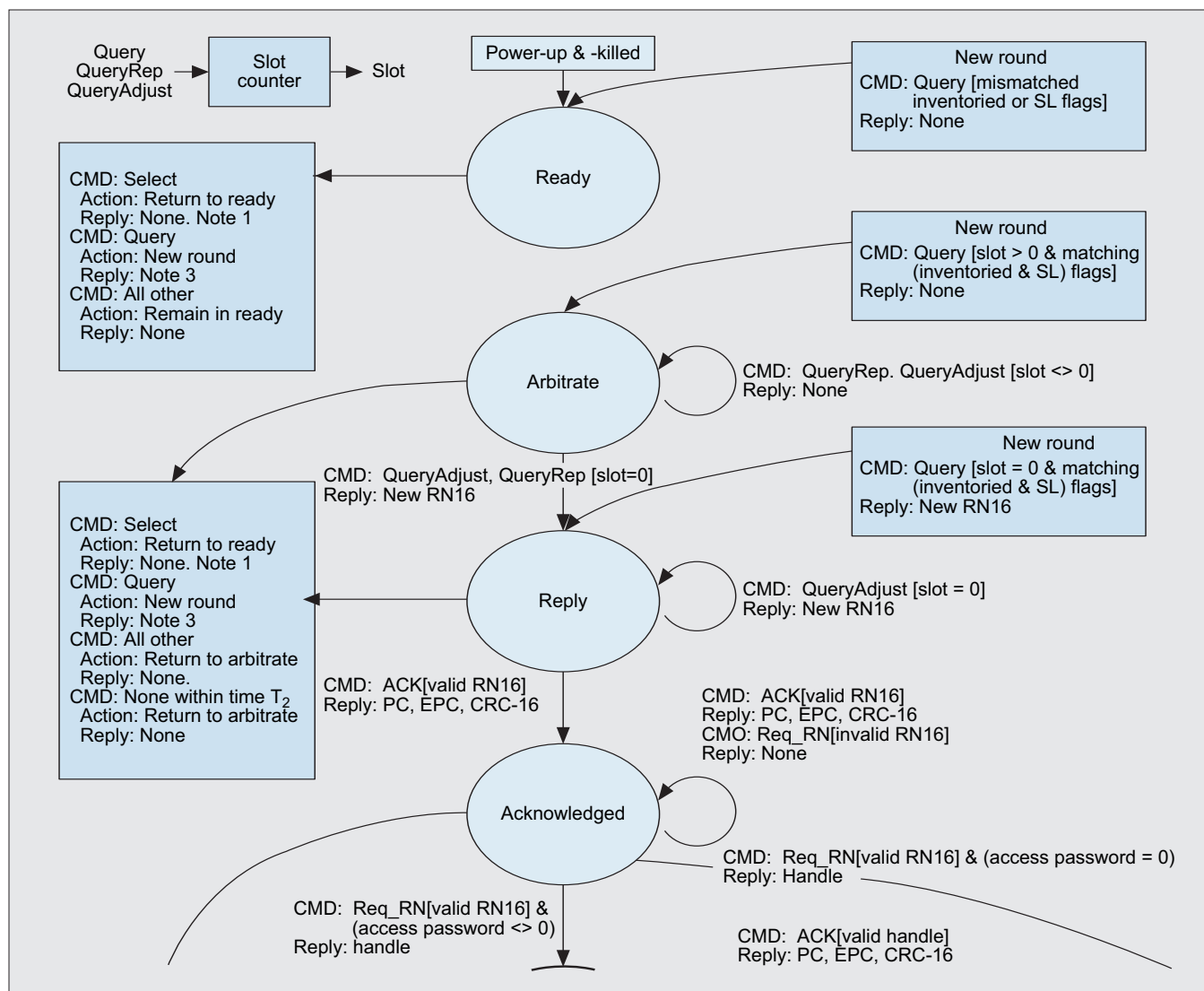
Tags implement a 15-bit slot counter, which is used for collision arbitration. The slot counter is loaded with a random value when the tag receives a *Query* command. The *QueryAdjust* and *QueryRep* commands can also modify the values of the slot counter.

Tags have seven states including ready, arbitrate, reply, acknowledged, open, secured, and killed. The ready state is a holding state for energized tags that are neither killed nor participating in an inventory round. The arbitrate state is a holding state for tags that are participating in an inventory round but whose slot counters hold non-zero values. Tags in the reply state backscatter the appropriate reply. A tag in the acknowledged state transitions to other states based on the command received. Tags in the open state can execute all access commands except *Lock*. Tags in the secured state can execute all access commands. The *Kill* command puts the tag in the killed state, in which it does not respond to any further interrogator commands. This state is not reversible. Figure 2 shows a partial state diagram of the tag. The state transitions introduced by some of the commands are shown.

Tag Memory — ISO 18000 Part 6C tags contain memory segmented into four memory banks. Figure 3 shows the logical memory map of the tag. The memory banks are for user memory, tag identifier (TID) memory, the EPC memory, and

Command	Code	Length (bits)	Mandatory?
QueryRep	00	4	Yes
Ack	01	18	Yes
Query	1000	22	Yes
QueryAdjust	1001	9	Yes
Select	1010	> 44	Yes
NAK	11000000	8	Yes
ReqRN	11000001	40	Yes
Read	11000010	> 57	Yes
Write	11000011	> 58	Yes
Kill	11000100	59	Yes
Lock	11000101	60	Yes
Access	11000110	56	No
BlockWrite	11000111	> 57	No
BlockErase	11001000	> 57	No

■ **Table 1.** ISO Part 6C command codes.



■ Figure 2. Partial state diagram of ISO Part 6C tag from the EPCGlobal Class 1 Generation 2 document [4].

a reserved memory. The user memory can be used for user-specific data storage. The TID memory contains the TID and may contain other vendor- or tag-specific data. The EPC memory contains a CRC-16, the protocol control (PC) bits, and the EPC that identifies the type of object to which the tag will be attached. The reserved memory contains the kill and access passwords. The access password is a 32-bit value that must be issued to put the tag in the secured state. The kill password is a 32-bit value that is used to permanently disable the tag. The interrogator uses the *Write*, *BlockWrite*, and *BlockErase* commands to edit the memory.

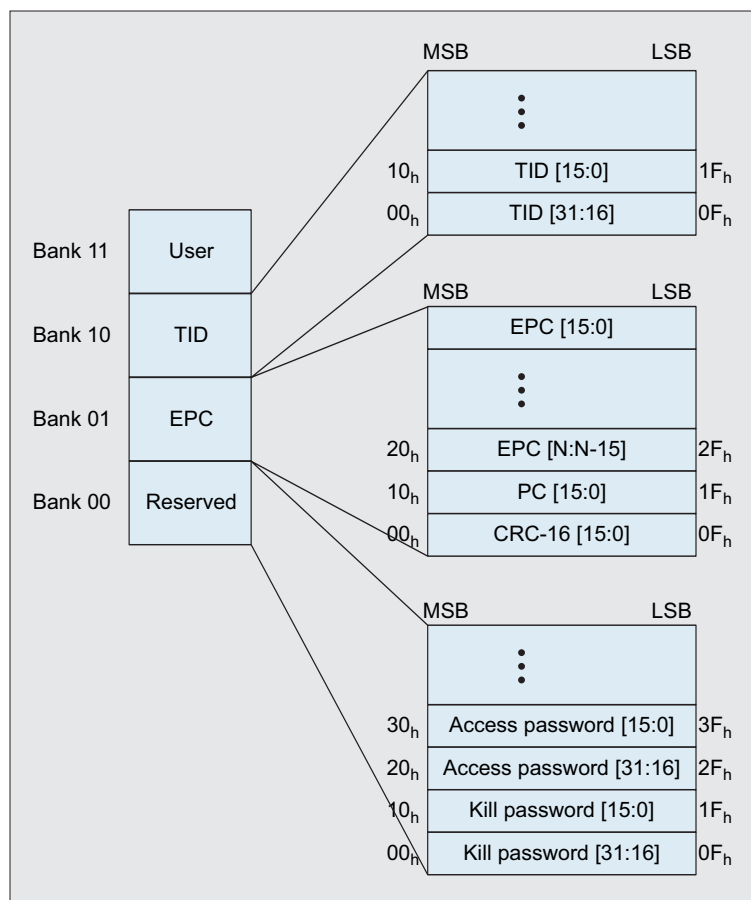
Security Features — The interrogator can lock or unlock each individual area of memory. This includes the access password, kill password, EPC memory bank, TID memory bank, and user memory bank. Tags must be in their secure state to use the lock command. Locking read and write protects the passwords and write protects the other memory banks. The interrogator can also *permalock* the lock status for a password or memory bank so that it is unchangeable. Permalock bits, once asserted, cannot be deasserted. This prevents the memory from being altered maliciously. Another security feature in the 18000 Part 6C standard is the Kill command. It can permanently deactivate the tag when the interrogator issues the correct kill password. After this is executed, the tag no longer

responds to interrogator commands.

IMPLEMENTING ISO 18000 PART 7 AND PART 6C

The ISO 18000 Part 7 and Part 6C protocols represent opposite ends of the complexity spectrum for tag implementations. As shown in the protocol overviews earlier, ISO 18000 Part 7 is analogous to a RISC style processor, and ISO 18000 Part 6C is analogous to a CISC style processor. RISC processors are based on a small number of simple instructions, but require a large instruction count for an application. CISC processors have a number of complex instructions that allow an application to require a much lower instruction count. Typically, embedded processors that target reduced power as a metric are RISC style processors with the prime example of the ARM processor family. General-purpose processors, particularly for desktop computers where power is only a concern for heat dissipation, the processor architectures are more CISC with vector and very long instruction word (VLIW) processing engines to augment the processor.

The commands or primitives issued by the interrogator request that the tag perform a set of actions. The specifications of these commands widely vary from ISO Part 7 to Part



■ **Figure 3.** Logical memory map of an ISO Part 6C tag.

6C. We have developed a RFID compiler that automatically generates RFID tag controller code targeting a microprocessor or hardware device based on a high-level description of the commands to be implemented [6, 7]. The flow of the RFID compiler is specified in Fig. 4.

To automate the generation of the tag controller for a prototype implementation, we implement the primitives as simple assembly-like instructions called *RFID macros*. The command code of each primitive is a unique field or opcode that serves as a command's identifier. Each RFID macro description contains a relatively short character string corresponding to the specific name of the primitive, a number indicating how many bits are used to represent the opcode of this particular primitive, as well as the distinct number corresponding to the value of the opcode. Additionally, a set of operands that correspond to the fields from each primitive and response is included.

The user specifies the tag behavior in a programming language such as ANSI C. To simplify the user interaction, the RFID parser generates a template for the response behavior indicating where the user must specify such custom behavior. Any C language constructs (conditionals, loops, etc.) can be added (or left unchanged) by the user to check the values of the fields of the incoming primitive and to specify the values of the fields of the response. The RFID compiler, *rfcc*, generates the tag controller code based on the input RFID macros and tag behavior. The final output of *rfcc* is either a C program or VHDL hardware description. The C program is used for the microprocessor-based controller, and the VHDL hardware description is used for a hardware-based controller such as an application-specific integrated circuit (ASIC) or FPGA implementation.

For the VHDL hardware description, the completed behavior in ANSI-C needs to be converted to synthesizable

Primitives	Dynamic power (mW)	Area (μm^2)
Query	0.06752	11,767.46
Collection	0.06308	10,944.83
10 primitives	0.06495	11,232.92

■ **Table 3.** Power and energy results for implementing query, collection, and 10 ISO Part 7 primitives (inclusive of collection) as a 0.16 μm ASIC. ASIC area is in $100 \mu\text{m}^2$. Dynamic power is in milliwatts. Quiescent power < $0.4 \mu\text{W}$.

hardware code. This is done by feeding it into the SuperCISC compiler [15-17] and combining it with the automatically generated VHDL in the RFID compiler. The resulting synthesizable VHDL is synthesized, mapped, placed, and routed for the hardware-based controller using commercially available tools. More details of the implementation of the RFID compiler may be found in [7].

RESULTS

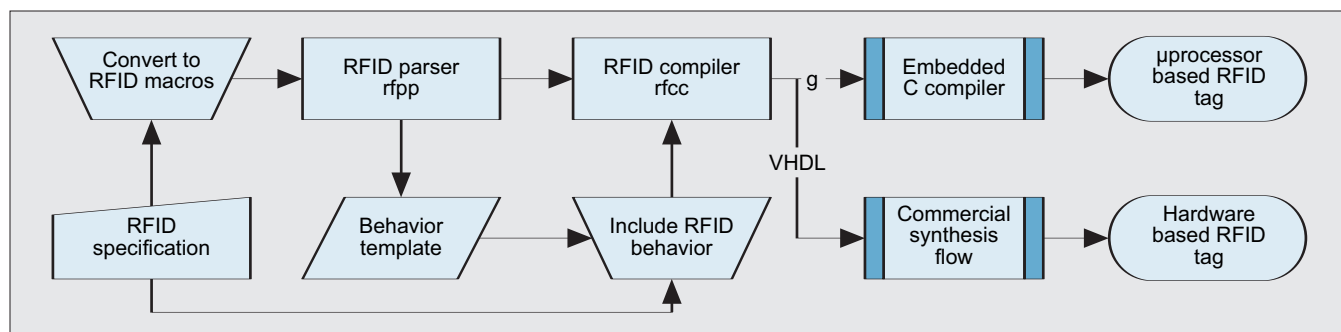
To understand the complexity of the two standards, the RFID compiler has been used to implement commands from ISO 18000 Part 7 and Part 6C.¹ A representative command, Query, was selected from ISO 18000 Part 6C and has been implemented in hardware and targeted to a 0.16 mm standard cell-based ASIC technology from Oki Semiconductor. For comparison, the *Collection* command, which realizes similar functionality to ISO 18000 Part 7, was implemented in hardware and targeted to the same ASIC process. Table 3 shows the power and area results for implementing these two commands. The Query command is much larger and higher power consuming than the Collection command. We also compared the Query command with 10 primitives from ISO 18000 Part 7. As shown in Table 3, the Query command is still larger and morepower consuming than 10 primitives from ISO 18000 Part 7.

CONCLUSIONS

This article explores and compares the two prevalent UHF RFID standards, the ISO 18000 Part 6C standard for passive tags and the ISO 18000 Part 7 standard for active tags. The communication primitives of ISO 18000 Part 6C are significantly different and more complex than those of ISO 18000 Part 7. ISO 18000 Part 6C relies on intermediate storage and storage of state at several points during each communication operation. The states and keys of the target device are used to facilitate operations such as tag singulation, collision arbitration, and security encoding.

To analyze the complexity of the respective tag designs, we used a RFID tag specification methodology and design automation flow described earlier. The compiler generates RFID tag controller code, targeting a microprocessor or hardware device, based on a description of the commands to be implemented. We compare the ASIC implementations of the ISO 18000 Part 6C Query command and the ISO 18000 Part 7

¹ For implementation of some primitives, a different version of the RFID compiler was used that does not synthesize the hardware from a C description.



■ Figure 4. RFID compilation flow.

Collection command, which have similar functionality. The Query implementation is not only much larger and consumes much higher power than the Collection command implementation, it also requires more power than 10 combined ISO 18000 Part 7 commands.

The existence of multiple keys and intermediate states and the complexity of primitives in the ISO 18000 Part 6C protocol presents challenging design problems compared to the simpler ISO 18000 Part 7 protocol. One solution is to examine the needs of the ISO 18000 Part 6C protocol and explore modifications to the protocol that lead to simpler implementations. A second possibility is to examine ways to decompose the CISC-like primitives in the standard to create a more RISC-like implementation.

REFERENCES

- [1] ANSI, "ANSI NCITS 236:2001 Standard Specification," 2002.
- [2] IISO/IEC FDIS 18000-7:2004(e), 2004.
- [3] IISO/IEC FDIS 18185-1:2006, 2006, under development.
- [4] EPCglobal, Inc., "EPC Radio-Frequency Identity Protocols: Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz–960 MHz," v. 1.0.9, Jan. 2005.
- [5] IISO/IEC FDIS 18000-6:2004/amd 1:2006(e), 2006.
- [6] A. K. Jones et al., "Passive Active Radio Frequency Identification Tags (PART)," *Int'l. J. Radio Freq. Identification Technology and Application*, vol. 1, no. 1, 2006, pp. 52–73.
- [7] A. K. Jones et al., "An Automated, FPGA-Based Reconfigurable, Low-Power RFID Tag," *J. Microprocessors and Microsys.*, 2006.
- [8] D. Engels and S. Sarma, "The Reader Collision Problem," white paper MITAUTOID-WH-007, Auto-ID Center. Nov. 2001.
- [9] R. Want, "The Magic of RFID," Oct. 2004.
- [10] A. Juels, R. Rivest, and M. Szydlo, "The Blocker Tag: Selective Blocking of RFID Tags for Consumer Privacy," *Proc. 10th ACM Conf. Comp. and Commun. Sec.*, 2003, pp. 103–11.
- [11] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, "Strong Authentication for RFID Systems Using the AES Algorithm," *Proc. 6th Int'l Wksp. Cryptographic Hardware and Embedded Sys.*, 2004, pp. 357–70.
- [12] S. A. Delichatsios et al., "Albano Multidimensional UHF Passive RFID Tag Antenna Designs," *Int'l. J. Radio Freq. Identification Technology and Application*, vol. 1, no. 1, 2006, pp. 24–40.
- [13] G. Barber and E. Tsibertzopoulos, "An Analysis of Using EPCglobal Class-1 Generation-2 RFID Technology for Wireless Asset Management," *Proc. IEEE MILCOM '05*, Oct. 2005, pp. 245–51.
- [14] C. K. Harmon, "The Necessity for a Uniform Organization of User Memory in RFID," *Int'l. J. Radio Freq. Identification Technology and Application*, vol. 1, no. 1, 2006, pp. 41–51.
- [15] A. K. Jones et al., "Reducing Power while Increasing Performance with SuperCISC," *ACM Trans. Embedded Comp. Sys.*, vol. 5, no. 3, Aug. 2006, pp. 1–29.
- [16] R. Hoare et al., "Rapid VLIW Processor Customization for Signal Processing Applications Using Combinational Hardware Functions," *EURASIP J. Applied Signal Proc.*, vol. 2006, Article ID 46 472, 2006.
- [17] A. K. Jones et al., "A VLIW Processor with Hardware Functions: Increasing Performance while Reducing Power," *Trans. Circuits and Systems II*, 2006.

BIOGRAPHIES

SWAPNA DONTARAJU is a Ph.D. candidate in the Department of Electrical and Computer Engineering, University of Pittsburgh. She received her B.E. in electrical and electronics engineering from the Regional Engineering College, Trichy, in 2002. She received her M.S. in computer science and engineering from Pennsylvania State University, University Park, in 2004. Her

research interests include compilers for low-power RFID tags, electronic design automation, and hardware design.

SHENCHIH TUNG [S] is a Ph.D. candidate in the Department of Electrical Engineering at the University of Pittsburgh, Pennsylvania. He received his B.S. degree in electrical engineering from National Taiwan Ocean University in 1997. He received his M.S. degree in telecommunication from the Department of Information Science at the University of Pittsburgh in 2000. His research interests include parallel computing and architecture, multiprocessor systems on a chip, network-on-chip systems, FPGA design, and RFID design. He is a student member of the IEEE Computer Society.

ALEX K. JONES received his B.S. in physics from the College of William and Mary, Williamsburg, Virginia. He received his M.S. and Ph.D. degrees in 2000 and 2002, respectively, in electrical and computer engineering at Northwestern University. He is currently an assistant professor of electrical and computer engineering and computer science at the University of Pittsburgh. He was formerly a research associate in the Center for Parallel and Distributed Computing, and an instructor of electrical and computer engineering at Northwestern University. He is a Walter P. Murphy Fellow of Northwestern University, a distinction he was awarded twice. His research interests include compilation techniques for behavioral and low-power synthesis, embedded systems, RFID, and high-performance computing. He is the author of over 40 publications in these areas.

LEONID MATS [M] received his B.S. in electrical engineering, B.S. in computer science, and M.S. in electrical engineering from the University of Pittsburgh in 1997, 1997, and 2001, respectively. He is currently working toward a Ph.D. degree in electrical engineering at the University of Pittsburgh. He worked as a software engineer from 2001 through 2003 at NeoLinear Inc. (now Cadence), Pittsburgh, where he was responsible for the design and implementation of module generators for high-frequency CMOS devices. He was also responsible for the integration of the module generators as parameterized cells into the Cadence Virtuoso. He research interests are in antenna design, low-power circuit design techniques, RF front-ends for RFID tags, and microwave theory.

JUSTIN PANUSKI received his B.S. in electrical engineering from the University of Pittsburgh in 2006. While an undergraduate, as a member of the Co-Op program he worked for ArgonST on radar systems. He is currently pursuing his M.S. in electrical engineering at the University of Pittsburgh.

J. T. CAIN [F] received B.S., M.S., and PhD degrees from the University of Pittsburgh in 1964, 1966, and 1970. He is currently a professor of electrical engineering and computer engineering at the University of Pittsburgh. He has also been a visiting professor at the University of Karlsruhe, Germany. His current research interests are in the system-on-a-chip area with emphasis on embedded systems and RFID systems. In addition to his research, he has been an active contributor to and office holder in a variety of professional organizations including the IEEE (President, 1995) and the Computing Sciences Accreditation Board (CSAB) as co-founder and 1988–1989 President. He is a Fellow of CSAB, and a member of the ACM, ASEE, and Sigma Xi.

MARLIN H. MICKLE [F'97] (mickle@pitt.edu) received B.S.E.E., M.S.E.E., and Ph.D. degrees from the University of Pittsburgh in 1961, 1963, and 1967, respectively. Currently, his research is focused on RFID and related technologies focused on autonomous device characterization. He has held engineering positions with Westinghouse and IBM as well as program director at the National Science Foundation. He was the 1988 recipient of the Systems Research and Cybernetics Award from the International Institute for Advanced Studies in Systems Research and Cybernetics; the Carnegie Science Center 2005 Award for Excellence in Corporate Innovation; and University Innovator Awards in 2005 and 2006.