Proceedings of the 28th IEEE
EMBS Annual International Conference
New York City, USA, Aug 30-Sept 3, 2006

Generic Controller Dedicated to Telemetry-Controlled Microsystems

Amir M. Sodagar, *Member, IEEE*, Kensall D. Wise, *Fellow, IEEE*, Khalil Najafi, *Fellow, IEEE*

***Abstract*—This paper introduces a generic controller designed for telemetry-controlled microsystems. This controller receives a data packet through a serial link carrying a command word and the associated data, and is capable of generating a variety of control/timing signals according to the definition of the received command. The flexible *microprogrammed* architecture of the controller allows for defining the commands functions in an on-chip mask-programmable read-only memory.**

I. Introduction

Telemetry-controlled implantable microsystems are of increasing interest in a wide variety of applications, ranging from biomedical implantable microsystems to tele-supervised sensing/monitoring devices. Unidirectional or bidirectional communication of these systems with a host or supervising unit is made possible using a wirelss link. This link allows for the transfer of a serial stream of digital data packets required to control the operation of the microsystem from the supervising unit to the microsystem. The data packets sent to these microsystems are usually designed in the most compact form in order to both efficiently use the communication bandwidth and decrease the probability of the occurrence of data error. The data packets are usually asynchronously transferred and hence, a start pulse is sent ahead of the main data bits informing the microsystem that the data is about to arrive. To be safe against the errors expected in wireless data transfer, system designers usually prefer to embed some parity bits in the data packet. At the receiver side, the microsystem is expected to be able to do a parity check and prevent any kind of false operation due to the probable errors.

While some microsystems have employed general purpose commercial controllers [1] & [2], which are not expected to be the best choice for highly-integrated low-power micro-systems, but most designers prefer to design and fabricate their own special-purpose controller in order to consume the least possible power and area [3]-[8]. Except for a few cases in which the controller has been implemented using gate arrays [9], [10], almost everybody has chosen the full-custom approach.

Manuscript received April 24, 2006. This work was supported primarily by the National Institutes of Health (NIH) under Contract HHSN265200423631C.

Amir M. Sodagar is with the Center for Wireless Integrated Microsystems (WIMS), University of Michigan, Ann Arbor, Michigan 48109, USA. Telephone: (734)763-5217; Fax: (734)647-2342; e-mail: asodagar@umich.edu.

Kensall D. Wise is with the Center for Wireless Integrated Microsystems (WIMS), University of Michigan, Ann Arbor, Michigan 48109, USA. e-mail: wise@umich.edu.

Khalil Najafi is with the Center for Wireless Integrated Microsystems (WIMS), University of Michigan, Ann Arbor, Michigan 48109, USA. e-mail: wise@umich.edu

From the standpoint of the controller architecture, in some cases the controller has been formed by connecting the required blocks as needed [6], [7]. A second category uses a well-formed architecture based on Finite State Machine method [3], [8].

This paper introduces a generic controller with a microprogrammed architecture [11], dedicated to telemetry-controlled microsystems.

II. Structure & Functionality

Shown in Figure 1 is the block diagram of the controller designed with a micro-programmed architecture that allows for fully-programmable commands and functions.

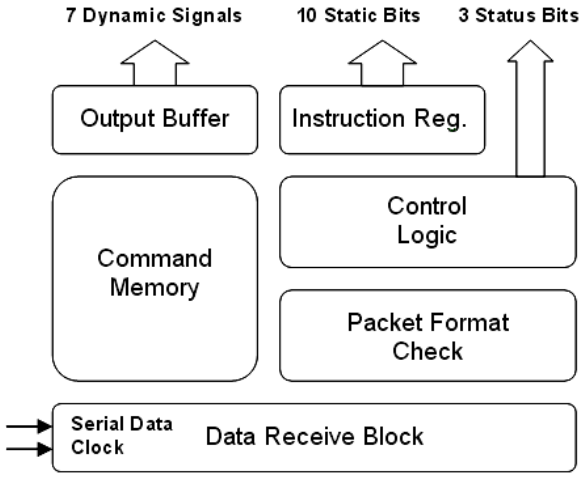


Figure 1 Block diagram of the controller

There are four free-form commands defined for the current version of the controller: CMD0~CMD3, whose function can be defined using a mask-programmable micro-program memory. The controller is capable of:

- receiving a data packet through a serial digital link,
- performing "Packet Format Check" including detecting a start pulse ahead of the information, and parity check,
- executing the received command, which can result in generating static and dynamic signals according to the information stored in the Command Memory, and
- providing status signals indicating the occurrence of parity errors and the start and the end of the execution of the received command.

It should be added that execution of a command can result in providing both fixed (static) and time-varying (dynamic) signals, in both parallel and serial fashions, and with both internal (programmed) and external (received through the serial link) origins. The dynamic signals change

state (switch between 0 and 1) synchronously with the master clock.

The above-mentioned features have been considered as the minimum capabilities that such a controller needs to have. Moreover, this controller is available as a core, and can be equipped with an additional application-specific interface to generate additional signals as required.

III. Format of the Data Packet

In general, an incoming data packet consists of a start pulse informing the system that a set of information is about to arrive, one field to transfer the pre-defined commands, probably one or more fields assigned to some addresses and/or data that come with some of the commands, and the parity bits that accompany the information.

To be more specific, the data packet is defined as follows: To make a Start Pulse, the serial link stays high for 4 periods of the master clock. Following the Start Pulse, 10 bits of information along with 3 parity bits arrive. Each bit of the data packet is defined to stay high or low for one period of the master clock (non-return to zero) and synchronous with it. General format for this data packet is shown in Figure 2. The four commands defined for the controller, CMD0~CMD3, are encoded by B0 and B1. B2-B9 can be used as data fields as needed and P0-P2 are even parity bits for B0-B3, B4-B6, and B7-B9, respectively.

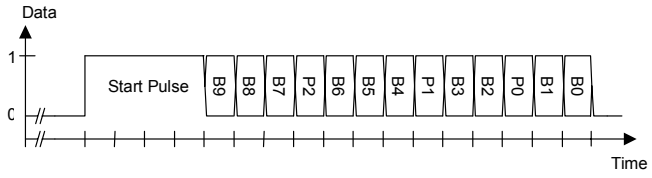


Figure 2 Format of the data packet

Figure 3 shows the internal structure of the mask-programmable read-only memory (ROM); called Command ROM (CROM), which contains the definitions of the command functions. Layout of a part of the two-dimensional matrix of data storage devices is shown in Figure 4, illustrating how the memory is programmed in mask level. Wherever a "1" is written in the memory, the gate of the storage transistor is connected to the associated Word Line. However in a regular deign there is no need to do anything specific to store zeros, but here the gate of the storage transistor is tied to V_{SS} in order to store a "0". This ensures us that no mistake will be made during "memory read" cycle caused by any unwanted voltage induction or noise in telemetry-powered applications.

IV. Programmable Commands

To accommodate the command definitions, 512 bits of memory space is partitioned into four 16-linex8-column blocks, each assigned to one of the commands. This allows each command to generate predefined high-low transition patterns on up to 7 Dynamic Signals (DS0~DS6) during the Execution Cycle which takes 16 cycles of the master clock. In other words, the memory block assigned to each command, in fact, contains the associated transitions patterns for the dynamic signals when the command is executed.

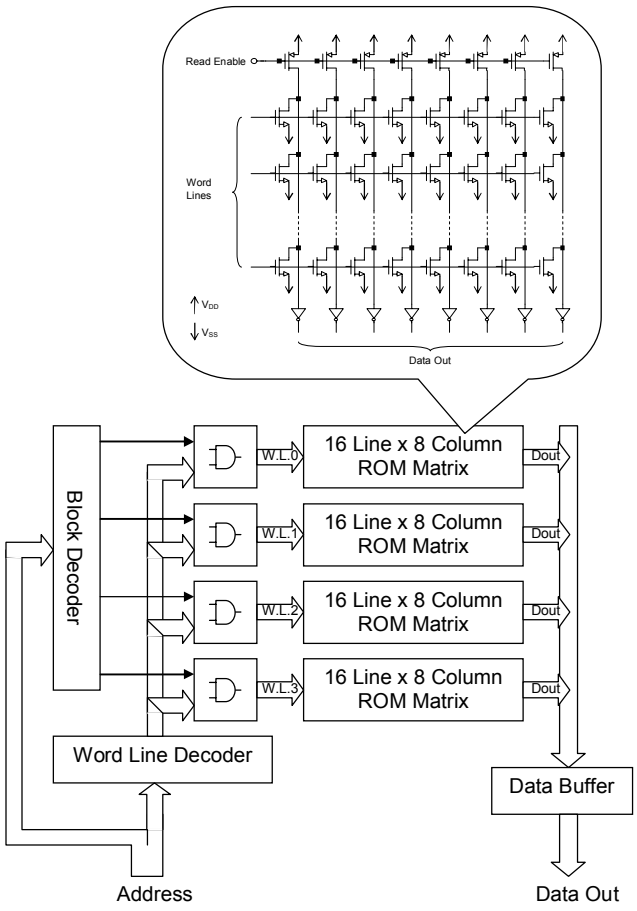


Figure 3 Microporgram Memory

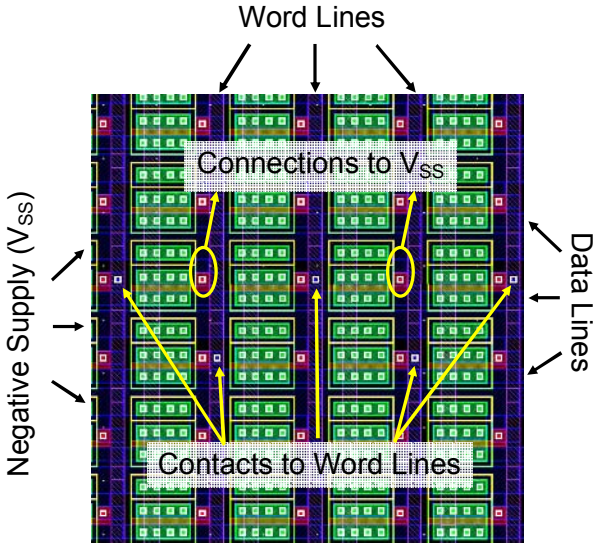


Figure 4 Layout of a part of the two-dimensional matrix of data storage devices

V. Design for Low Power Consumption

In many applications, telemetry-controlled microsystems are expected to be telemetry powered as well. Hence, it would be important to design the controller for low power consumption. However, it should be noted that when talking about power minimization in such applications, robustness of operation and noise immunity are still the more important concerns that should not be sacrificed for the sake of power. This leads us to the conventional static CMOS logic family as our optimal choice for standard digital logic, which has zero DC power dissipation and can have acceptable noise margin [12]. To implement the microprogram memory, a pseudo-NMOS logic family is chosen to keep the chip area small. To avoid non-necessary DC current draw by the memory cells that store 1 when the memory is not read, the matrix's current sources can be turned off using the "Read Enable" signal, as shown in Figure 3.

VI. Tests

This section deals with the application of the designed controller in a telemetry-controlled inductively-powered multi-channel neural recording implantable microsystem, along with the demonstration of the associated tests on the fabricated controller. The system consists of four multi-channel neural recording active probes, a mixed-signal neural processing chip, and a bidirectional telemetry chip. Equipped with an application-specific interface and implemented on the bidirectional telemetry chip, the controller is in charge of the administration of the whole system based on the received commands and data. All the four controller commands are used, three of them accompanied by one or two addresses and a data word, which should be serially delivered to some specific blocks.

Figure 5 shows the photomicrograph of the fabricated controller along with the added interface on the bidirectional telemetry chip. Fabricated using a 1.5µm double-metal double-poly standard CMOS process, the controller and the added interface occupy approximately 7 mm^2 of silicon area, and are expected to be scaled down to less than 1mm^2 when fabricated in a 0.5µm process in the near future.

For the tests demonstrated in this section, a serial bit stream is fed into the controller as the received data packet along with the associated clock. The four commands in this application are called STSEL for *site selection* on the active probes, THSET for *threshold setting* on the neural processing chip, and SNMOD and MNMOD to set the whole system in the *Scan Mode* or the *Monitor Mode*, respectively.

Part (a) of Figure 6 shows an incoming data packet, and also the execution of the received command (STSEL). Shown in part (b) is the operation of the controller when two consecutive commands, THSET and MNMOD, are received. This test is performed to both demonstrate the controller's function in response to a THSET command and also examine the signals, which are supposed to be used in the two consecutive commands to see whether any unwanted spike occurs when the controller switches from one command to the next.

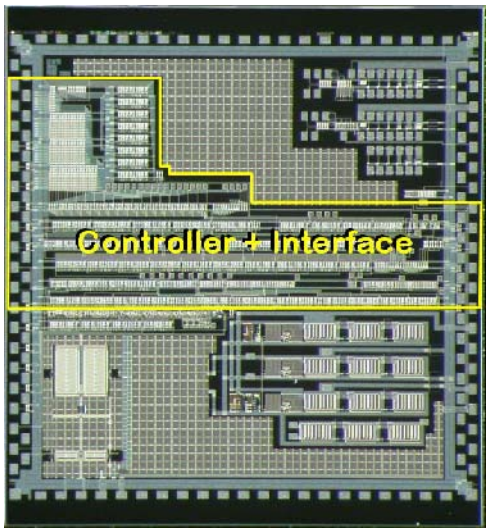
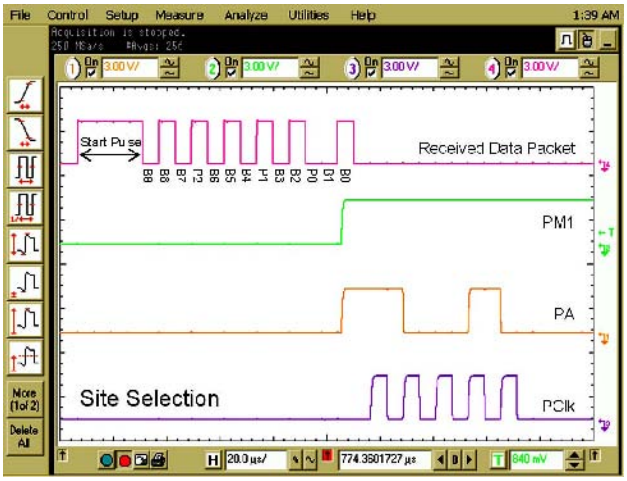
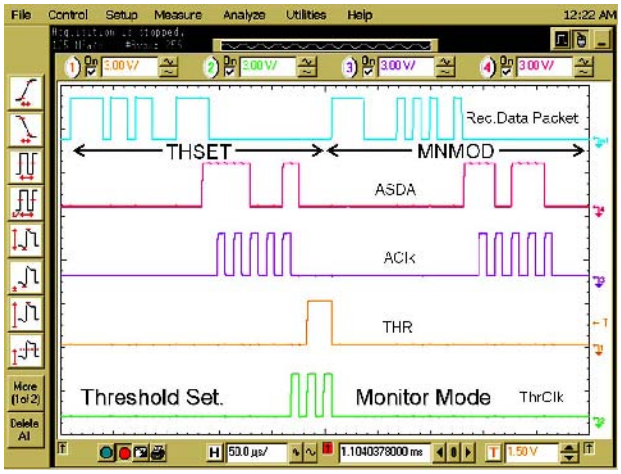


Figure 5 Photograph of the fabricated controller



(a)



(b)

Figure 6 (a) STSEL Command (b) THSET Command Followed by MNMOD

Reaction of the controller to a parity error in the received data packet is shown in Figure 7. The received command data is exactly the same as that of Figure 6(b) except that a "1" is missing in the second command. As can be seen, the second command is not accepted at all, and all the signals stay at the same levels as before, as if no second command has received. TABLE 1 summarizes the controller's specifications

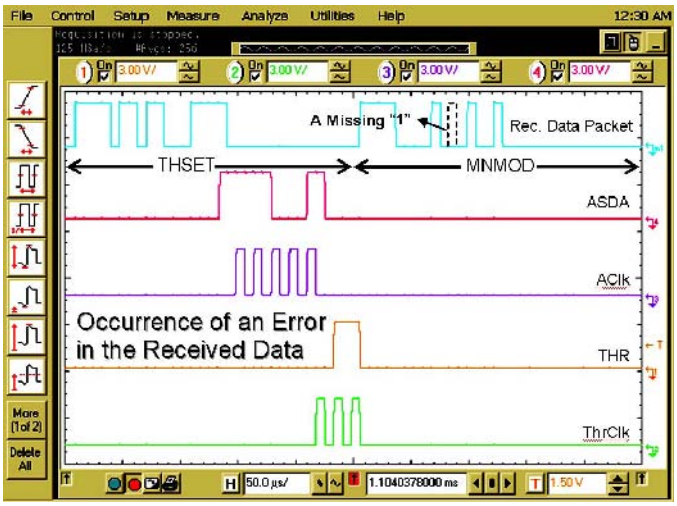


Figure 7 The same simulation as that of Figure 6(b), but a parity error occurs in the data packet for the MNMOD

TABLE 1
SPECIFICATION SUMMARY

Specification	Value
Supply Voltage	3V
Fabrication Process	1.5µm N-Well CMOS
Circuit Area	7mm^2
Nominal Clock Frequency	2MHz
Incoming Data Packet:	
Total No. of Bits	*17 bits*
Start Pulse	*4 bits*
Command Code	*2 bits*
Data and Address	*8 bits*
Parity	*3 bits*
On-Chip Command ROM	512 bits
Output Signals:	
Dynamic Signals (for up to 16 clock cycles)	*7*
Static Signals	*10*

REFERENCE

[1] Farshchi S., Mody I., Judy, J.W., "A TinyOS-based wireless neural interface," Proceedings of 26th Annual International Conference of the Engineering in Medicine and Biology Society (EMBC 2004), Vol. 2, pp. 4334 – 4337.

[2] S. Farshchi, et al, "A TinyOS-Based Wireless Neural Sensing, Archiving, and Hosting System," Proceedings of the 2nd International IEEE EMBS Conference on Neural Engineering, March 16-19, 2005, pp. 671 – 674.

[3] J. A. Von-Arx, and K. Najafi, "A Wireless Single-Chip Telemetry-Powered Neural Stimulation System," Proceedings of the 1999 IEEE Int'l Solid-State Circuits Conference (ISSCC'99), pp. 214-215.

[4] Ba, A., Sawan, M., "Integrated Programmable Neuro-Stimulator to Recuperate the Bladder Functions," IEEE CCECE 2003. Canadian Conference on Electrical and Computer Engineering, Vol. 1, pp. 147 – 150, May 2003.

[5] Coulombe J., Gervais J.-F., Sawan M., "A cortical stimulator with monitoring capabilities using a novel 1 Mbps ASK data link," Proceedings of the 2003 Int'l Symposium on Circuits & Systems (ISCAS '03), Vol. 5, pp. 53 – 56, May 2003.

[6] B. Gosselin, et al, "Multi-channel wireless cortical recording: circuits, system design and assembly challenges," IEEE International Workshop on Biomedical Circuits and Systems, pp. S1/7/INV - S1/79-12, Dec. 2004.

[7] A. D. DeHennis, K. D. Wise, "A wireless microsystem for the remote sensing of pressure, temperature, and relative humidity," Journal of MicroElectroMechanical Systems, Vol. 14, No. 1, Feb. 2005, pp. 12 – 22.

[8] K. Arabi, M. Sawan, "Electronic design of a multichannel programmable implant for neuromuscular electrical stimulation," IEEE Transactions on Rehabilitation Engineering, Vol. 7, No. 2, June 1999, pp. 204 – 214.

[9] M. Sawan, et al, "Computerized transcutaneous control of a multichannel implantable urinary prosthesis," IEEE Transactions on Biomedical Engineering, Vol. 39, No. 6, June 1992, pp. 600 – 609.

[10] S. Boyer, et al, "Implantable selective stimulator to improve bladder voiding: design and chronic experiments in dogs," IEEE Transactions on Rehabilitation Engineering, Vol. 8, No. 4, Dec. 2000 pp. 464 – 470.

[11] Andrew S. Tanenbaum, *Structured Computer Organization*, Prentice Hall, 5th Edition, 2005.

[12] Ken Martin, *Digital Integrated Circuit Design*, Oxford University Press, 1999.