Backscatter RFID Chip

ECE 429 course project

Author: Dan White and contributors

Date: 2016-04-07 22:16

Version: v0.1-11-gf1189e9-**dirty**

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC-2119].

NOTE: The .rst version of this document SHALL be considered the canonical version, the .pdf is merely a convenience. Discrepancies MUST resolve to the .rst version.

1.2 Radio-frequency identification

Radio-frequency identification (RFID) is a wireless communication scheme where a Reader transmits a strong signal towards a device (Tag) and the Tag responds by sending back data. This communication from the Tag to Reader is accomplished by the Tag varying the impedance of its antenna. The energy impinging on the Tag's antenna is absorbed and reflected ("scattered") in some proportion depending on the antenna's impedance characteristics.

If the Tag changes its antenna impedance rapidly, the back-scattered energy will change also, resulting in the scattered wave having double-sideband modulation components in addition to the original frequency. The Reader can detect these sidebands and demodulate the data that the Tag sent.

The most common method for modulating these back-scattered sidebands is to vary the frequency of the impedance changes between two frequencies. Bits are

[RFC-2119] https://www.ietf.org/rfc/rfc2119.txt therefore able to be assigned to each unique frequency. In other words, the Tag switches the antenna impedance between two levels (usually open- / short-circuit) at frequency #1 or frequency #2 depending on the current bit to be sent.

Such a communication method requires extremely little power consumption on the Tag side of the link. With careful design, the Tag can even extract enough energy from the incoming signal from the Reader to power itself and operate the antenna switch.

1.3 Synchronous Serial Communication

The two most common interfaces to connect peripheral devices to a central processor both use serial data connections, SPI and I2C. It is possible, and even somewhat common, to find devices which are compatible with both formats using the same pins.

- SPI : Serial Peripheral Interface bus
- I2C : Inter-Integrated-Circuit bus

The details and timing diagrams for each of these formats are easily found on the internet.

2 Project Specifications

The project for ECE 429 is to design and layout an integrated circuit in the On Semiconductor C5N 0.5um CMOS process that implements the major subsystems of an RFID tag. A complete design would be capable of transmitting arbitrary data on programmable backscatter channel frequencies in the 915 MHz ISM band and also possibly in the 2.4 GHz ISM band.

2.1 Processor interface specification

The processor interface to this chip SHALL be via an SPI slave port. From the view of the controlling processor, the device is a bank of up to 128 registers of 8-bits each which may be written to or read from. The chip datasheet MUST specify the implemented address locations and the meaning of reads and/or writes to those addresses. Writes to an unimplemented address SHOULD have no effect. Reads of unimplemented register addresses MAY return meaningless data and SHOULD be ignored by the controlling processor.

The chip datasheet SHALL clearly specify the SPI mode in terms of CPOL and CPHA as used in reference [WP-SPI].

There SHALL be two commands accepted by the device: *read-register* and *write-register*. For an SPI bus transaction, these commands are encoded in the first and most-significant bit of the first byte sent to the chip. *Read-register* is encoded as a 1 while *write-register* is encoded as a 0 value. The following two tables describe the protocol used for register read and write commands.

Table 1: SPI register write transaction. Raddr is the 7-bit register address and Rdata is the 8-bit register data to be stored.

Pin	**byte0**	**byte1**
Bit #:	76543210	76543210
MOSI	0[Raddr]	[Rdata]
MISO	xxxxxxxx	xxxxxxxx

Table 2: SPI register read transaction. Raddr is the 7-bit register address and Rdata is the 8-bit register data stored at that address.

Pin	**byte0**	**byte1**
Bit #:	76543210	76543210
MOSI	1[Raddr]	xxxxxxxx
MISO	xxxxxxxx	[Rdata]

Implementation of a serial data interface compatible with both SPI and I2C is OPTIONAL.

The chip's I2C device address MUST be within the range of valid addresses according to the I2C specification. The least-significant bits of the address MAY be pin-programmable, i.e. zero or more pins MAY be used to set the last address bits while the prefix bits are hard-coded to some valid value.

Register read or write commands SHALL follow the same format as the SPI-based protocol except the first byte is the I2C standard device address and read/write bit. Since the *read* or *write* command is specified in the least-significant bit of the

[WP-SPI] https://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus first transaction byte, the chip SHALL ignore the most-significant bit of the register address byte. The following tables describe the chip's 3-byte I2C protocol.

Table 3: I2C register write

Pin	**byte0**	**byte1**	**byte2**
Bit #:	`76543210`	`76543210`	`76543210`
`SDA`	`<Daddr>0`	`x<Raddr>`	`<8-data>`

Table 4: I2C register read

Pin	**byte0**	**byte1**	**byte2**
Bit #:	`76543210`	`76543210`	`76543210`
`SDA`	`<Daddr>1`	`x<Raddr>`	`<8-data>`

2.1.1 Protocol references

https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi
https://learn.sparkfun.com/tutorials/i2c
http://www.i2cchip.com/mix_spi_i2c.html
http://www.i2c-bus.org/
http://www.nxp.com/documents/user_manual/UM10204.pdf
Other links: http://wavedrom.com/
http://www.timing-diagrams.com/

2.2 Switch mapper

The switch mapper translates the mode of operation (FSK, or QAM) into appropriate antenna switch states. Switch states are translated as `switch[x] = 0`: NMOS off, and `switch[x] == 1`: NMOS on.

In FSK mode (`mode == 0`), the input *fmod* is directly passed to `switch[0]` while the other switches remain off. For QAM mode (`mode == 1`), the 2-bit input *symbol[1:0]* determines which single switch is on and the *fmod* input is ignored.

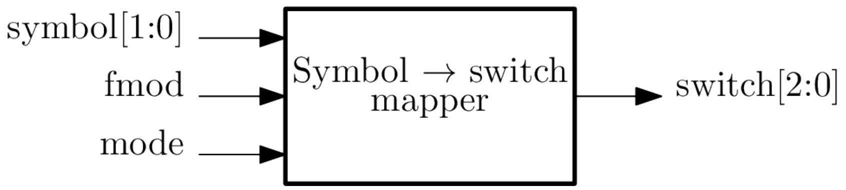

Figure 1: Switch state mapping block diagram. See the table "Symbol to antenna switch mapping table" for the decoding.

Table 5: Symbol to antenna switch mapping table.

mode	**fmod**	**symbol[1:0]**	**switch[2:0]**
0	0	XX	000
0	1	XX	001
1	X	00	000
1	X	01	001
1	X	10	010
1	X	11	100

2.3 Numerically-controlled oscillator (NCO)

A numerically-controlled oscillator forms the basis of the programmable backscatter frequency control for both channel selection and frequency-shift-keying (FSK) modulation. The NCO SHALL use two 8-bit frequency control words, *fcw0[7:0]* and *fcw1[7:0]*, which are applied to a multiplexer whose output is selected by the state of *fsel* The current state of the phase accumulator register and the selected frequency control word SHALL be added, ignoring the carry-out, and used to set the next state of the phase accumulator register. This causes the accumulator to increment its state by the value of the selected *fcw* at each clock cycle.

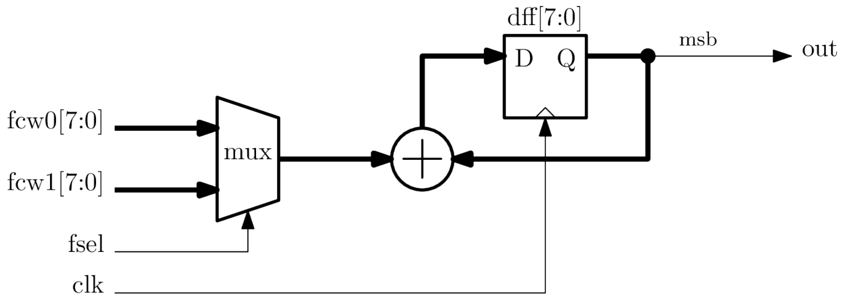

Figure 2: Numerically-controlled oscillator diagram and signals.

Only the most-significant bit of the phase accumulator SHALL used as the output signal, which is a square wave at an average frequency of:

$$f_{out} = \frac{fcw}{256} f_{clk}$$

The smallest change in average output frequency for the NCO is given by:

$$f_{res} = \frac{f_{clk}}{256}$$

The duty cycle is not guaranteed to be 50% -- the high and low times may vary by ± 1 clock period. See reference [WP-NCO] for more information about NCO output characteristics.

2.4 Antenna switches

These switch various impedances in parallel with the antenna to vary its net impedance and thence backscatter magnitude/phase.

Three N-type switches SHALL be used

2.5 Charge pump

Accepts antenna input and outputs semi-regulated DC.

[WP-NCO] https://en.wikipedia.org/wiki/Numerically_controlled_oscillator