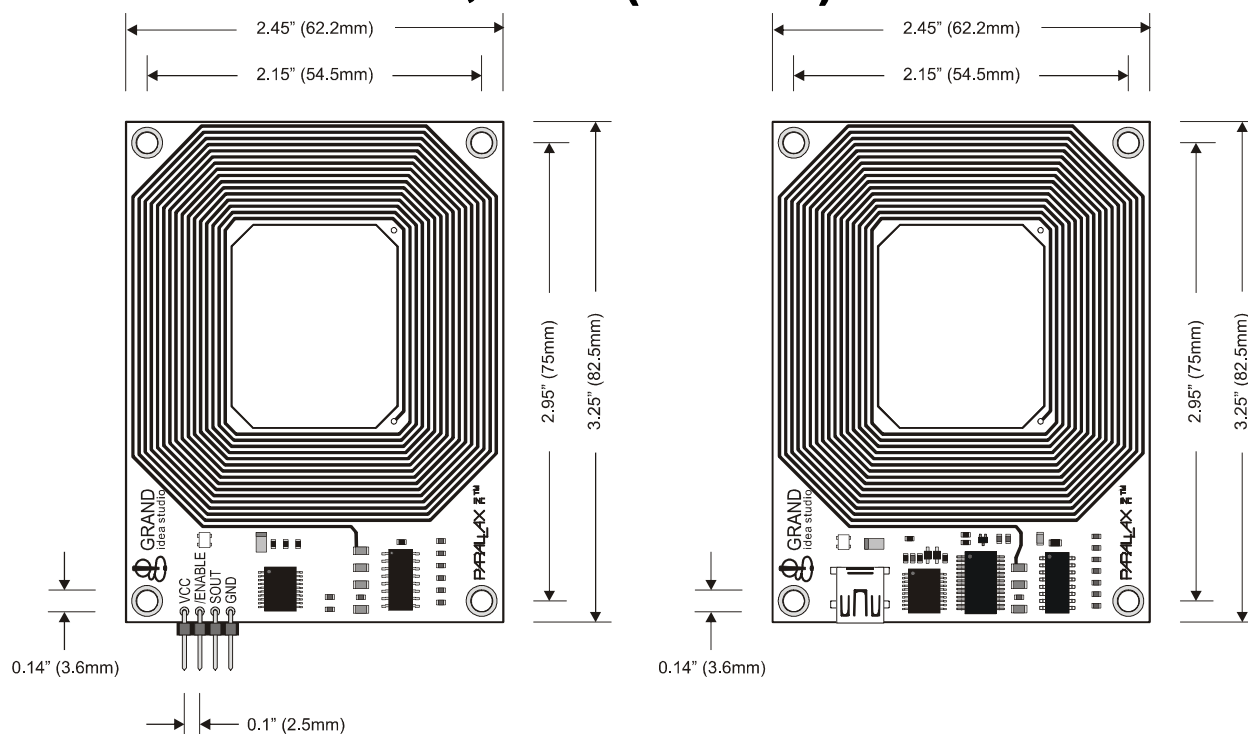


RFID Card Reader, Serial (#28140) RFID Card Reader, USB (#28340)



Introduction

Designed in cooperation with Grand Idea Studio (<http://www.grandideastudio.com/>), the Parallax Radio Frequency Identification (RFID) Card Readers provide a low-cost solution to read passive RFID transponder tags up to 4 inches away. The RFID Card Readers can be used in a wide variety of hobbyist and commercial applications, including access control, automatic identification, robotics navigation, inventory tracking, payment systems, and car immobilization. The RFID Card Reader is available in a TTL version for microcontroller interfacing as well as a USB version for direct connection to a PC, Mac or Linux computer.

Features

- Low-cost method for reading passive RFID transponder tags
- 2400 bps serial interface
- Enable input allows module to be enabled/disabled by software
- USB version connects directly to PC, Mac or Linux machine and requires no additional wiring
- Bi-color LED for visual indication of status

RFID Compatibility

The Parallax RFID Card Reader works exclusively with the EM Microelectronics-Marin SA EM4100-family of passive read-only transponder tags at 125 kHz. A variety of different tag types and styles exist with the most popular made available from Parallax. Each transponder tag contains a unique identifier (one of 2^{40} , or 1,099,511,627,776 possible combinations) that is read by the RFID Card Reader and transmitted to the host via a simple serial interface.

Communication Protocol

When the RFID Card Reader is active and a valid RFID transponder tag is placed within range of the activated reader, the unique ID will be transmitted as a 12-byte printable ASCII string serially to the host in the following format:

Start Byte (0x0A)	Unique ID Digit 1	Unique ID Digit 2	Unique ID Digit 3	Unique ID Digit 4	Unique ID Digit 5	Unique ID Digit 6	Unique ID Digit 7	Unique ID Digit 8	Unique ID Digit 9	Unique ID Digit 10	Stop Byte (0x0D)
----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	-----------------------	---------------------

The start byte and stop byte are used to easily identify that a correct string has been received from the reader (they correspond to a line feed and carriage return characters, respectively). The middle ten bytes are the actual tag's unique ID. For example, for a tag with a valid ID of 0F0184F07A, the following bytes would be sent: \$0A, \$30, \$46, \$30, \$31, \$38, \$34, \$46, \$30, \$37, \$41, \$0D

All communication is 8 data bits, no parity, 1 stop bit, and least significant bit first (8N1). The baud rate is configured for 2400 bps, a standard communications speed supported by most any microprocessor or PC, and cannot be changed. The Parallax RFID Card Reader initiates all communication. Output from the serial RFID Card Reader TTL level non-inverted. The USB version of the RFID Card Reader handles serial communications through the VCP (virtual COM port) driver. This allows easy access to the serial data stream from any programming language that can open a COM port.

Interface Details

The RFID Card Reader is activated via the /ENABLE line. When the RFID Card Reader is powered and the /ENABLE line is pulled HIGH, the module will be inactive (standby mode) and the LED will be GREEN. When the /ENABLE line is pulled LOW, the RFID Card Reader enter its active state and enable the antenna to interrogate for tags. The current consumption of the module will increase dramatically when the module is active. On the serial version of the RFID Card Reader the /ENABLE line is directly accessible via the 4-pin header. On the USB version this line is controlled by the COM port DTR line. Enabling DTR will activate the RFID Card Reader (LED Red) while disabling DTR will deactivate the RFID Card Reader (LED Green).

A visual indication of the state of the RFID Card Reader is given with the on-board LED. When the module is successfully powered-up and is in an idle state, the LED will be GREEN. When the module is in an active state and the antenna is transmitting, the LED will be RED. The LED can be used to visually indicate to a user that the tag has been read successfully by deactivating the reader briefly when the tag has been read.

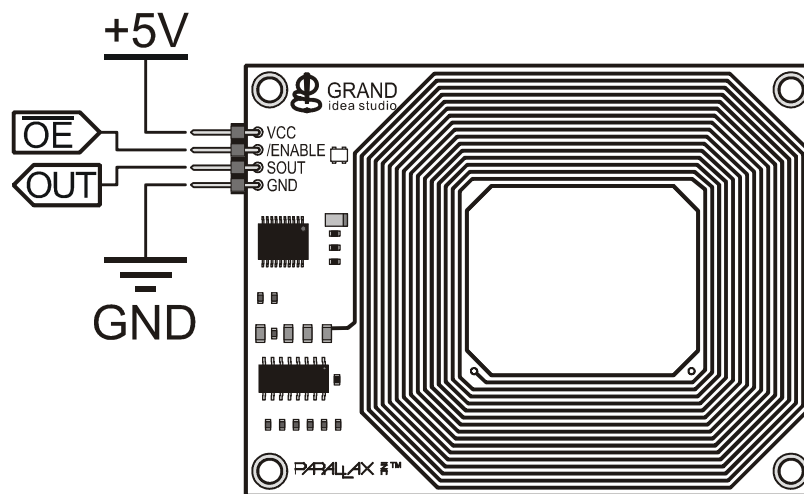
The face of the RFID tag should be held parallel to the front or back face of the antenna (where the majority of RF energy is focused). If the tag is held sideways (perpendicular to the antenna) you may have difficulty getting the tag to be read. Only one transponder tag should be held up to the antenna at any time. The use of multiple tags at one time will cause tag collisions and confuse the reader. The tags available in the Parallax store have a read distance of approximately 3 inches. Actual distance may vary slightly depending on the size of the transponder tag and environmental conditions of the application.

Connecting and Testing (Serial Version)

The Parallax RFID Card Reader serial version can be integrated into any microcontroller design using only four connections (VCC, /ENABLE, SOUT, GND). Use the following circuit for connecting the Parallax RFID Card Reader to the BASIC Stamp microcontroller.

Pin	Pin Name	Type	Function
1	VCC	P	System power, +5V DC input.
2	/ENABLE	I	Module enable pin. Active LOW digital input. Bring this pin LOW to enable the RFID reader and activate the antenna.
3	SOUT	O	Serial Output TTL-level interface, 2400bps, 8 data bits, no parity, 1 stop bit.
4	GND	G	System ground. Connect to power supply's ground (GND) terminal.

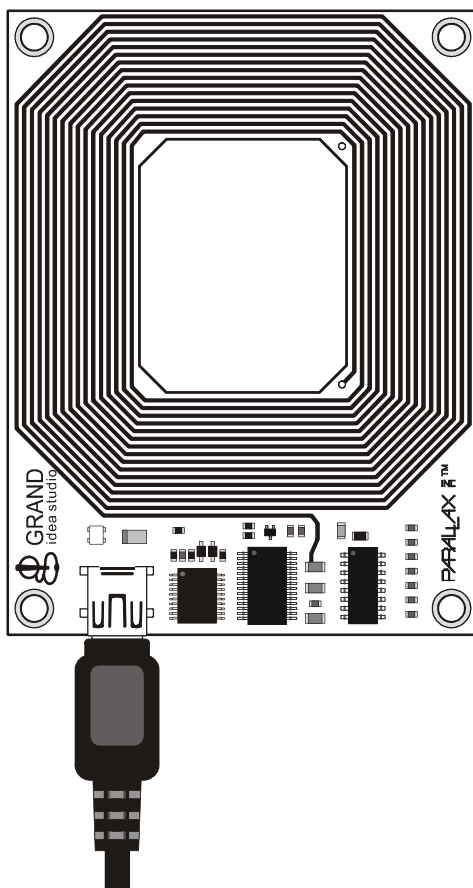
Note: Type: I = Input, O = Output, P = Power, G = Ground



Connecting and Testing (USB Version)

The Parallax RFID Card Reader USB can be connected to any PC, Mac or Linux computer with a USB port and the appropriate FTDI USB drivers installed. The drivers should be installed prior to connecting the hardware to your computer. The drivers for PC can be obtained from the Parallax website by going to the Support tab and selecting Downloads/FTDI USB Drivers. Other drivers, including Mac and Linux can be downloaded from the FTDI website at the following URL. <http://www.ftdichip.com/Drivers/VCP.htm> When the RFID Card Reader USB is connected to a computer with the appropriate drivers a COM port is automatically assigned to it. This COM port can be accessed by any program or programming language you choose that can open a COM port @ 2400 bps (8N1). Parallax has released a free sample program (including source code) for reading the tags in Windows XP/Vista. This program and VB.net source code can be downloaded from the RFID Card Reader product page by going to our website and typing in the product number 28340. You can also use the BASIC Stamp Editor DEBUG window to see tag data. Checking the DTR box in the toolbar will enable the RFID Card Reader to send tag data to the display.

Line	Port Function	Function
RX	Serial Receive	Serial data from RFID Card Reader USB
DTR	Data Terminal Ready	Activate/De-activate RFID Card Reader USB



Interference

The Parallax RFID Card Reader, like many RF devices, may experience RF noise in its frequency range. This may cause the reader to transmit a spurious tag response when no tag is near the unit. This will not affect most uses of the RFID Card Reader. To completely prevent spurious responses, it is recommended to simply read two responses in a row within a given amount of time (e.g. 1 second) to ensure that you are reading a valid tag and not a "tag" generated by noise.

DC Characteristics

At $V_{CC} = +5.0V$ and $T_A = 25^{\circ}C$ unless otherwise noted

Parameter	Symbol	Test Conditions	Specification			Unit
			Min.	Typ.	Max.	
Supply Voltage	V_{CC}	---	4.5	5.0	5.5	V
Supply Current, Idle	I_{IDLE}	---	---	10	---	mA
Supply Current, Active	I_{CC}	---	---	100	---	mA
Input LOW voltage	V_{IL}	$+4.5V \leq V_{CC} \leq +5.5V$	---	---	0.8	V
Input HIGH voltage	V_{IH}	$+4.5V \leq V_{CC} \leq +5.5V$	2.0	---	---	V
Output LOW voltage	V_{OL}	$V_{CC} = +4.5V$	---	---	0.6	V
Output HIGH voltage	V_{OH}	$V_{CC} = +4.5V$	$V_{CC} - 0.7$	---	---	V

Absolute Maximum Ratings

Condition	Value
Operating Temperature	$-40^{\circ}C$ to $+85^{\circ}C$
Storage Temperature	$-55^{\circ}C$ to $+125^{\circ}C$
Supply Voltage (V_{CC})	$+4.5V$ to $+5.5V$
Ground Voltage (V_{SS})	0V
Voltage on any pin with respect to V_{SS}	$-0.3V$ to $+7.0V$

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

RFID Technology Overview

Material in this section is based on information provided by the RFID Journal (www.rfidjournal.com).

Radio Frequency Identification (RFID) is a generic term for non-contacting technologies that use radio waves to automatically identify people or objects. There are several methods of identification, but the most common is to store a unique serial number that identifies a person or object on a microchip that is attached to an antenna. The combined antenna and microchip are called an "RFID transponder" or "RFID tag" and work in combination with an "RFID reader" (sometimes called an "RFID interrogator").

An RFID system consists of a reader and one or more tags. The reader's antenna is used to transmit radio frequency (RF) energy. Depending on the tag type, the energy is "harvested" by the tag's antenna and used to power up the internal circuitry of the tag. The tag will then modulate the electromagnetic

waves generated by the reader in order to transmit its data back to the reader. The reader receives the modulated waves and converts them into digital data.

There are two major types of tag technologies. "Passive tags" are tags that do not contain their own power source or transmitter. When radio waves from the reader reach the chip's antenna, the energy is converted by the antenna into electricity that can power up the microchip in the tag (known as "parasitic power"). The tag is then able to send back any information stored on the tag by reflecting the electromagnetic waves as described above. "Active tags" have their own power source and transmitter. The power source, usually a battery, is used to run the microchip's circuitry and to broadcast a signal to a reader. Due to the fact that passive tags do not have their own transmitter and must reflect their signal to the reader, the reading distance is much shorter than with active tags. However, active tags are typically larger, more expensive, and require occasional service. The RFID Card Reader is designed specifically for low-frequency (125 kHz) passive tags.

Frequency refers to the size of the radio waves used to communicate between the RFID system components. Just as you tune your radio to different frequencies in order to hear different radio stations, RFID tags and readers must be tuned to the same frequency in order to communicate effectively. RFID systems typically use one of the following frequency ranges: low frequency (or LF, around 125 kHz), high frequency (or HF, around 13.56 MHz), ultra-high frequency (or UHF, around 868 and 928 MHz), or microwave (around 2.45 and 5.8 GHz).

There really is no such thing as a "typical" RFID tag. The read range of a tag ultimately depends on many factors: the frequency of RFID system operation, the power of the reader, environmental conditions, physical size of the tags antenna and interference from other RF devices. Balancing a number of engineering trade-offs (antenna size v. reading distance v. power v. manufacturing cost), the Parallax RFID Card Reader's antenna was designed with a specific inductance and "Q" factor for 125 kHz RFID operation at a tag read distance of around 4 inches.

BASIC Stamp[®] 1 Program

The following code examples read tags from a RFID Card Reader and compare the values to known tags (stored in an EEPROM table).

```
' =====
'
' File..... RFID.BS1
' Purpose.... RFID Tag Reader / Simple Security System
' Author..... (c) Parallax, Inc. -- All Rights Reserved
' E-mail..... support@parallax.com
' Started....
' Updated.... 07 FEB 2005
'
' {$STAMP BS1}
' {$PBASIC 1.0}
'
' =====

' -----[ Program Description ]-----
'
' Reads tags from a Parallax RFID reader and compares to known tags (stored
' in EEPROM table). If tag is found, the program will disable a lock.

' -----[ Revision History ]-----
```

```

' -----[ I/O Definitions ]-----

SYMBOL  Enable          = 0          ' low = reader on
SYMBOL  RX              = 1          ' serial from reader
SYMBOL  Spkr            = 2          ' speaker output
SYMBOL  Latch           = 3          ' lock/latch control

' -----[ Constants ]-----

SYMBOL  LastTag          = 2          ' 3 tags; 0 to 2

' -----[ Variables ]-----

SYMBOL  tag0             = B0          ' RFID bytes buffer
SYMBOL  tag1             = B1
SYMBOL  tag2             = B2
SYMBOL  tag3             = B3
SYMBOL  tag4             = B4
SYMBOL  tag5             = B5
SYMBOL  tag6             = B6
SYMBOL  tag7             = B7
SYMBOL  tag8             = B8
SYMBOL  tag9             = B9

SYMBOL  tagNum           = B10         ' from EEPROM table
SYMBOL  pntr             = B11         ' pointer to char in table
SYMBOL  char             = B12         ' character from table

' -----[ EEPROM Data ]-----

Tags:
  EEPROM ("0F0184F20B")          ' valid tags
  EEPROM ("0F01D9D263")
  EEPROM ("04129C1B43")
  EEPROM ("0000000000")          ' space for other tags
  EEPROM ("0000000000")

' -----[ Initialization ]-----

Reset:
  HIGH Enable          ' turn of RFID reader
  LOW Latch            ' lock the door!

' -----[ Program Code ]-----

Main:
  LOW Enable          ' activate the reader
  SERIN RX, T2400, ($0A) ' wait for header
  SERIN RX, T2400, tag0, tag1, tag2, tag3, tag4 ' get tag bytes
  SERIN RX, T2400, tag5, tag6, tag7, tag8, tag9
  HIGH Enable          ' deactivate reader

Check_List:
  FOR tagNum = 0 TO LastTag          ' scan through known tags
    pntr = tagNum * 10 + 0 : READ pntr, char ' read char from DB
    IF char <> tag0 THEN Bad_Char ' compare with tag data

```

```

pntr = tagNum * 10 + 1 : READ pntr, char
IF char <> tag1 THEN Bad_Char
pntr = tagNum * 10 + 2 : READ pntr, char
IF char <> tag2 THEN Bad_Char
pntr = tagNum * 10 + 3 : READ pntr, char
IF char <> tag3 THEN Bad_Char
pntr = tagNum * 10 + 4 : READ pntr, char
IF char <> tag4 THEN Bad_Char
pntr = tagNum * 10 + 5 : READ pntr, char
IF char <> tag5 THEN Bad_Char
pntr = tagNum * 10 + 6 : READ pntr, char
IF char <> tag6 THEN Bad_Char
pntr = tagNum * 10 + 7 : READ pntr, char
IF char <> tag7 THEN Bad_Char
pntr = tagNum * 10 + 8 : READ pntr, char
IF char <> tag8 THEN Bad_Char
pntr = tagNum * 10 + 9 : READ pntr, char
IF char <> tag9 THEN Bad_Char
GOTO Tag_Found                                ' all match -- good tag

Bad_Char:
NEXT

Bad_Tag:
SOUND Spkr, (25, 80)                          ' groan
PAUSE 1000
GOTO Main

Tag_Found:
DEBUG #tagNum, CR                             ' for testing
HIGH Latch                                    ' remove latch
SOUND Spkr, (114, 165)                        ' beep
LOW Latch                                     ' restore latch
GOTO Main

END

```

BASIC Stamp® 2 Program

The following code examples read tags from a RFID Card Reader and compare the values to known tags (stored in an EEPROM table).

```

' =====
'
' File..... RFID.BS2
' Purpose.... RFID Tag Reader / Simple Security System
' Author..... (c) Parallax, Inc. -- All Rights Reserved
' E-mail..... support@parallax.com
' Started....
' Updated.... 07 FEB 2005
'
' {$STAMP BS2}
' {$PBASIC 2.5}
'
' =====

' -----[ Program Description ]-----
'
' Reads tags from a Parallax RFID reader and compares to known tags (stored
' in EEPROM table). If tag is found, the program will disable a lock.

```

```

' -----[ Revision History ]-----

' -----[ I/O Definitions ]-----

Enable      PIN      0      ' low = reader on
RX           PIN      1      ' serial from reader
Spkr        PIN      2      ' speaker output
Latch       PIN      3      ' lock/latch control

' -----[ Constants ]-----

#SELECT $STAMP
  #CASE BS2, BS2E, BS2PE
    T1200     CON      813
    T2400     CON      396
    T4800     CON      188
    T9600     CON      84
    T19K2     CON      32
    TMidi     CON      12
    T38K4     CON      6
  #CASE BS2SX, BS2P
    T1200     CON      2063
    T2400     CON      1021
    T4800     CON      500
    T9600     CON      240
    T19K2     CON      110
    TMidi     CON      60
    T38K4     CON      45
  #CASE BS2PX
    T1200     CON      3313
    T2400     CON      1646
    T4800     CON      813
    T9600     CON      396
    T19K2     CON      188
    TMidi     CON      108
    T38K4     CON      84
#ENDSELECT

SevenBit     CON      $2000
Inverted     CON      $4000
Open         CON      $8000
Baud         CON      T2400

#SELECT $STAMP
  #CASE BS2, BS2E
    TmAdj     CON      $100      ' x 1.0 (time adjust)
    FrAdj     CON      $100      ' x 1.0 (freq adjust)
  #CASE BS2SX
    TmAdj     CON      $280      ' x 2.5
    FrAdj     CON      $066      ' x 0.4
  #CASE BS2P
    TmAdj     CON      $3C5      ' x 3.77
    FrAdj     CON      $044      ' x 0.265
  #CASE BS2PE
    TmAdj     CON      $100      ' x 1.0
    FrAdj     CON      $0AA      ' x 0.665
  #CASE BS2Px
    TmAdj     CON      $607      ' x 6.03
    FrAdj     CON      $2A       ' x 0.166
#ENDSELECT

```

```

LastTag          CON      3

#DEFINE __No_SPRAM = ($STAMP < BS2P)          ' does module have SPRAM?

' -----[ Variables ]-----

#IF __No_SPRAM #THEN
    buf          VAR      Byte(10)          ' RFID bytes buffer
#ELSE
    chkChar      VAR      Byte              ' character to test
#ENDIF

tagNum           VAR      Nib               ' from EEPROM table
idx              VAR      Byte              ' tag byte index
char             VAR      Byte              ' character from table

' -----[ EEPROM Data ]-----

Tag1             DATA    "0F0184F20B"      ' valid tags
Tag2             DATA    "0F01D9D263"
Tag3             DATA    "04129C1B43"

Name0            DATA    "Unauthorized", CR, 0
Name1            DATA    "George Johnston", CR, 0
Name2            DATA    "Dick Miller", CR, 0
Name3            DATA    "Mary Evans", CR, 0

' -----[ Initialization ]-----

Reset:
    HIGH Enable  ' turn of RFID reader
    LOW Latch    ' lock the door!

' -----[ Program Code ]-----

Main:
    LOW Enable    ' activate the reader
    #IF __No_SPRAM #THEN
        SERIN RX, T2400, [WAIT($0A), STR buf\10] ' wait for hdr + ID
    #ELSE
        SERIN RX, T2400, [WAIT($0A), SPSTR 10]
    #ENDIF
    HIGH Enable    ' deactivate reader

Check_List:
    FOR tagNum = 1 TO LastTag          ' scan through known tags
        FOR idx = 0 TO 9              ' scan bytes in tag
            READ (tagNum - 1 * 10 + idx), char ' get tag data from table
            #IF __No_SPRAM #THEN
                IF (char <> buf(idx)) THEN Bad_Char ' compare tag to table
            #ELSE
                GET idx, chkChar        ' read char from SPRAM
                IF (char <> chkChar) THEN Bad_Char ' compare to table
            #ENDIF
        NEXT
    GOTO Tag_Found                    ' all bytes match!

```

```

Bad_Char:                                ' try next tag
    NEXT

Bad_Tag:
    tagNum = 0
    GOSUB Show_Name                      ' print message
    FREQOUT Spkr, 1000 */ TmAdj, 115 */ FrAdj ' groan
    PAUSE 1000
    GOTO Main

Tag_Found:
    GOSUB Show_Name                      ' print name
    HIGH Latch                          ' remove latch
    FREQOUT Spkr, 2000 */ TmAdj, 880 */ FrAdj ' beep
    LOW Latch                           ' restore latch
    GOTO Main

END

' -----[ Subroutines ]-----

' Prints name associated with RFID tag

Show_Name:
    DEBUG DEC tagNum, ": "
    LOOKUP tagNum,
        [Name0, Name1, Name2, Name3], idx ' point to first character
    DO
        READ idx, char                     ' read character from name
        IF (char = 0) THEN EXIT             ' if 0, we're done
        DEBUG char                         ' otherwise print it
        idx = idx + 1                      ' point to next character
    LOOP
    RETURN

```