

## iCE65 as MS Pro Interface

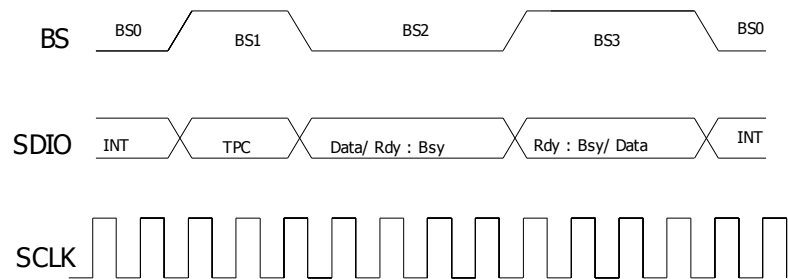
### Overview

Memory Stick is a removable flash memory card, which are used as storage media for a portable devices like digital cameras, digital music players, PDAs, cellular phones etc.. This design example illustrates the implementation of a Memory Stick interface using iCE65 FPGAs. As a Memory Stick interface, the iCE65 FPGA generates all the required interfacing signals and implements the functions that are required for such a system.

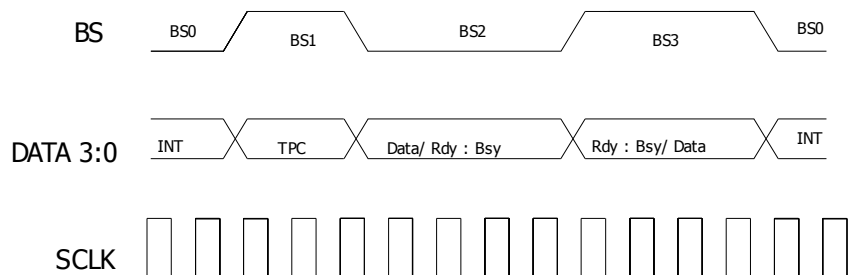
### Description

A Memory Stick consists of non-volatile flash based memory and a controller. Memory Stick family includes the Memory Stick PRO, a revision that allows greater maximum storage capacity and faster file transfer speeds; Memory Stick Duo, a small-form-factor version of the Memory Stick (including the PRO Duo); and the even smaller Memory Stick Micro (M2).

A Memory Stick Pro(MS Pro) can be controlled through three-wire half-duplex serial protocol, or through the six-wire parallel protocol. The Serial I/F provides to transfer data with three signal lines SCLK, BS and SDIO. The Parallel I/F provides a protocol to transfer data with six signal lines SCLK, BS and DATA[3:0]. A Host controller accesses the Registers and Buffer with command groups called TPC(Transfer Protocol Command). When a MS Pro is turned ON, it operates in the serial I/F mode and can be switched to the Parallel I/F mode with a TPC. It can also be switched from Parallel I/F mode to serial I/F mode with the TPC. Figure 1 and 2 shows the timing diagram of Serial mode interface and Parallel mode interface. Table 1 lists the pin description of MS Pro device.



**Fig 1: Serial Interface**



**Fig 2: Parallel Interface**

Pin	Direction	Functionality
sclk	Input	Clock Signal
ins	Output	MS Pro Insertion/Removal detector. This pin goes low when MS Pro inserted to the socket
data_ms[3:0]	Inout	Data Signal in Serial I/F. Data Signal 0 to 3 in Parallel I/F.
bs	Input	Bus State Signal. This signal operates in 4 modes. BS0 : Idle state. BS is held low. BS1 : TPC command Write state. BS held High. BS2 : BS held low. If data read operation, then BUSY/RDY status available on Data lines. If data write operation, then writes data on Data lines. BS3: BS Held High. If data write operation, then BUSY/RDY status available on Data lines. If data read operation, then data from MS Pro is available on Data lines.

**Table 1: MS Pro Pin Description**

## Implementation

MS Pro Interface connection diagram is shown in Figure 3. MS(Memory Stick)/MS-Pro Interface Module uses Parallel mode interface protocol for read/write data to MS Pro. MS IF(Interface) serializes the commands, data and communicates to the Memory Stick. MS IF interprets commands from MS Host controller as either read or write based on read or write bit in the TPC. Based on this control signal, the IF either writes the data or reads the data from MS.

### Data/Command read/write control

When chip select cs\_n asserted low, then command/data read/write operations are executed as follows:

- when "00" – No operation. MS Pro Stick deselected.
- when "01" - Command Write to MS IF TPC Register
- when "10" - Data Read from MS IF Data Register
- when "11" - Data Write to MS IF Data Register

Data is read/written in MSB byte first format.

### Command Register Contents(TPC)

When the command register is written, the communication protocol with the MS Pro device starts. The data transfer direction with the Memory Stick is determined from TPC[3]. When TPC[3] = 0, the read protocol is performed, otherwise the write protocol is performed. When the protocol starts, ms\_ready changes to '0' to indicate that protocol execution is underway. ms\_ready held 0 throughout the communication with the Memory Stick, and generates an interrupt at the end of communication by sending a High on ms\_ready pin. TPC Register[15:12] is the command part and TPC[9:0] provides the Transmit/Receive Data Size information to the MS Pro IF.

Table 2 below lists interface details of this Design Example.

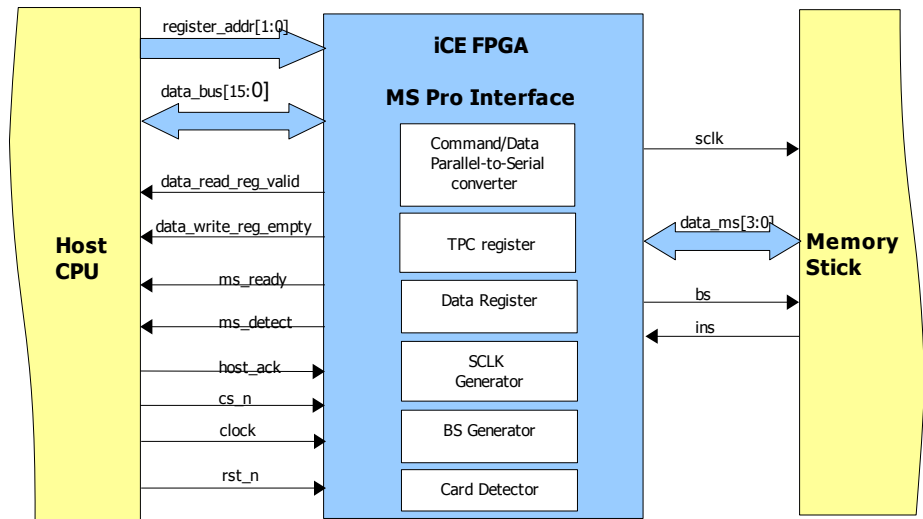
Table 3 below shows the post P&R resource utilization summary of this design when implemented on an iCE65 FPGA.

Pin	Direction	Description
cs_n	Input	Chip select signal generated by the host
rst_n	Input	Active low system reset
clock	Input	System clock
data_bus [15:0]	Inout	Data from/to cpu/host controller
register_addr[1:0]	Input	Selects the data or command register operation
data_read_reg_valid	Output	Goes high when data register is full. (For read operation )
data_write_reg_empty	Output	Goes high when data register is empty. (For write operation )
ms_detect	Output	MS card detection to CPU/HOST. Active High.
ms_ready	Output	Busy/ready status to CPU. Low when busy, High when idle.
host_ack	Input	Host sends a HIGH on this pin as soon as it acknowledges data_out_valid or data_reg_empty signals. If not acknowledged, IF suspends the communication by keeping sclk LOW
sclk	Output	Clock Signal generated, which is 1/4 <sup>th</sup> the system clock frequency
ins	Input	MS Pro Insertion/Removal detector. This pin goes low when MS Pro inserted to the socket
data_ms[3:0]	Inout	Data Signal 0 to 3 in Parallel I/F
bs	Output	Bus State Signal. This signal operates in 4 modes. BS0 : Idle state. BS is held low. BS1 : TPC command Write state. BS held High. BS2 : BS held low. If data read operation, then BUSY/RDY status available on Data lines. If data write operation, then writes data on Data lines. BS3: BS Held High. If data write operation, then BUSY/RDY status available on Data lines. If data read operation, then data from MS Pro is available on Data lines

**Table 2: Pin description of MS Pro Interface**

Device	Logic Cells	IO Cells
ICE65L04-UCB284	234	33

**Table 3: Resource Utilization**



**Fig 3: Block Diagram of MS Pro Interface**

## Conclusion

This design example demonstrates the implementation of a MS Pro interface using iCE FPGAs. iCE FPGA's very low power capabilities makes iCE FPGAs an obvious choice for implementing a MS Pro interface for battery operated compact and handheld devices like PDAs, cellular phones etc..

## Disclaimer

SiliconBlue is providing this document, design example, or information "as is." SiliconBlue does not ensure that this implementation or information is void of any claims of infringement. As a reader / implementer, you are responsible for obtaining any rights you may require for your implementation. SiliconBlue also does not warranty the fitness of this design example to be readily implemented in any specific context. It is your sole responsibility to validate this design example and its correctness while implementing it for your requirement.