NET+50 Errata

Part number/version: 90000283_L
Release date: February 2006
www.digi.com

For use in conjunction with:
NET+50/20M Hardware Reference
Part number/version: 90000281_A
Release date: September 2002

Making DEVICE NETWORKING *easy*™

Digi International
11001 Bren Road East
Minnetonka, MN 55343 U.S.A.
United States: +1 877 912-3444
Other locations: +1 952 912-3444
Fax: +1 952 912-4960
www.digi.com/support
www.digi.com

Contents

NET+50 Errata

This document contains information about NET+50 chip issues.

How to identify your NET+50 chip

The NET+50 is provided in two versions, each with its own part number:

BGA package

- 0137003 — Pb-based; last sold in September 2003
- 0137003LF — RoHS-compliant since September 2003

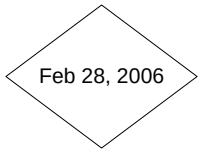
PQFP package

- 0136993LF — Pb-free; last sold in May 2005
- 1136993 — RoHs-compliant since May 2005

The part number appears on the surface of the chip.

NET+50 errata

The next sections describe the known errata for the NET+50 chip. Each section describes the problem, and in most cases, provides a workaround.

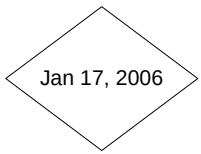


Serial port error in 7-bit mode

The NET+50's serial ports do not function correctly when configured in the 7-bit mode. The 7-bit transmitter will transmit 8-bit data, and the data in the receive buffer will be incorrect. The other modes — 8-bit, 6-bit, and 5-bit — all function correctly.

Workaround

None. Use the 8-bit, 6-bit, or 5-bit modes only.



UART CTS-related transmit data errors

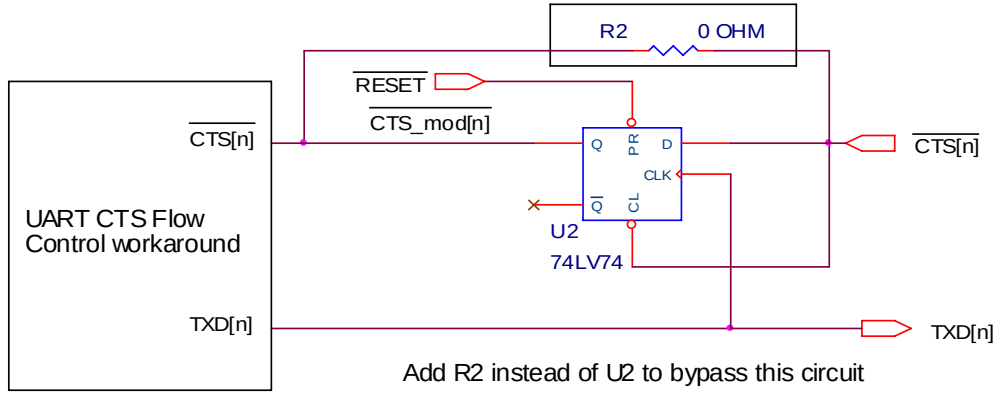
A problem occurs when the CTS flow control signal is de-asserted during the BCLK that begins processing a new character. This problem causes the previous character to be re-transmitted instead of getting the next character from the transmit FIFO.

Software workarounds

- Modify these bits in Serial Channel Control register A:
  - Set the CTSTX bit (bit 23) to 0 to disable hardware-controlled CTSTX.
  - Set the ERXCTS bit (bit 4) to enable the software CTS signal change interrupt.
  - Update the serial transmit ISR to handle the CTS signal change.
- Non-DMA mode: Because FIFO can hold up to eight lines of 4 bytes each, the maximum skid rate will be 32 characters.
- To reduce the maximum skid rate to eight characters, the software is changed to write 1 byte in each line.
- In DMA mode, the skid rate can be up to the number of bytes transmitted in 1 tick.
- The serial monitor thread is changed to handle the missing CTS interrupt.

Hardware workaround

For each UART, externally clock the CTS signal with the Txd_n signal to guarantee that CTS will not be seen de-asserting at the start of a character.



Timing violation with refresh logic

See the *NET+50/20M Hardware Reference*, Chapter 8, "Memory Controller Module."

The NET+50 refresh sequence for an SDRAM consists of three commands. This sequence cannot be changed by software. The sequence of commands is REFRESH, NOP, NOP, with each command consuming a single clock cycle. At the maximum operating frequency of 44.2368 MHz, the three commands take 67.8 ns to complete. This violates the auto refresh period of many SDRAM components.

Workaround

Use one of these options to prevent this problem:

- Select only those SDRAM components that specify an auto refresh period that is 67 ns or less. This timing parameter is always specified in manufacturers' documentation as **Trfc**.
- Run the NET+50 at a frequency no higher than 42.8 MHz. With a 18.432 MHz crystal, the PLL comes up following a hardware reset to the internal system clock (SYSCLK) at 44.2368 MHz. If you need to use a different frequency for SYSCLK, you must use a different external configuration; that is, a crystal with a different output frequency and/or an external oscillator.

CPU cache control does not operate with debugger interface

See the *NET+50/20M Hardware Reference,* Chapter 7, "Cache."

The NET+50 cache controller does not operate properly while interfacing with the ARM debugger interface. The ARM debugger interface cannot be used to read or write the contents of either the internal RAM or the cache RAM itself. When you use the ARM debugger to modify or view the contents of internal RAM, the memory becomes corrupted. This prevents setting firmware breakpoints for any code in the cache.

Workaround

Avoid using cache in debug mode.

CPU module mixed cache and internal RAM operation

See the *NET+50/20M Hardware Reference,* Chapter 7, "Cache."

The NET+50 cannot be used in an arbitrary mixed cache and internal RAM configuration. The internal high-speed 0-wait state memory in the CPU Module must be configured to use the lowest SET as RAM. Incorrect cache/RAM configurations cause instructions to remain stuck in the cache controllers; flushing does not occur.

Workaround

In CCR0 and CCR1, the sets can be enabled as shown here (where R = RAM and C = cache):

Sets	1 2 3 4
	C C C C
	R C C C
	R R C C
	R R R C
	R R R R

CPU module cache operation using 16-bit peripherals

See the *NET+50/20M Hardware Reference,* Chapter 7, “Cache.”

The NET+50 cache does not function at its peak performance specification when used to cache from 16-bit peripherals. The cache controller thrashes when loading the upper and lower parts of a 32-bit word from 16-bit memory, resulting in some loss of performance when compared to a theoretical cache working at full potential.

Workaround

None.

Chip select IDLE bit crash

Setting the IDLE bit in any non-DRAM chip select causes a crash when running code from DRAM.

Workaround

Do not set the IDLE bit in a non-DRAM chip select.

EFE module ERXBAD filters packets dribble nibble errors

See the *NET+50/20M Hardware Reference,* Chapter 10, “Ethernet Controller.”

Setting the ERXBAD bit in the Ethernet General Control Register causes packets with any of these conditions to be automatically filtered by the hardware:

- CRC error
- Dribble nibble condition
- Code errors

This configuration bit should not have been filtering packets with a Dribble Nibble Condition. These packets are still considered valid within the IEEE 802.3u standard.

Workaround

The ERXBAD bit in the Ethernet General Control Register should be set to 1 at all times.

The Ethernet Receive driver firmware can filter out packets with CRC errors or Coding errors using bits 10 and 8 of the Ethernet Receive Status word. The Ethernet Receive Status word is automatically written to the DMA Buffer Descriptor status field for each received Ethernet packet.

ENI/IEEE 1284 module HOST ECP operation

See the *NET+50/20M Hardware Reference,* Chapter 12, "MIC Controller Module."

IEEE 1284 HOST ECP Inbound does not work properly. The hardware does not wait for peripheral ACK* to be deasserted high before attempting to read the next character.

Workaround

None.

ENI PACK* signal intermittent hang

In an ENI bus cycle, the PACK* signal, as driven by the NET+ARM in acknowledgement of an externally driven PCS* active low at the start of a new cycle, has been observed to hang (stuck at active low) indefinitely, or until a terminating condition occurs (for example, PCS* going back to high). This problem manifests itself when PACK* is configured in RDY mode, and seems to be a random failure that may not be observed for many hours at a time, or at all.

Workaround

Synchronization of the external PCS* input with the negative edge of BCLK through a dual-rank flip-flop should ensure a proper hold time on the PCS* signal.

Flash burst read mode intermittent failure

When both cache and Flash Burst Read mode are used, intermittent failures (once in a 24-hour period, independent of data or environmental conditions) occur where the Flash Chip Select (CS0*) remains persistently asserted even though a non-Flash address (for example, SRAM activated by CS1*) operand is currently requested. As a result, bad data is read. Subsequent reads to that same address return the proper (SRAM) data.

Workaround

Disable Flash Burst mode if using the NET+ARM cache.

MAC module excessive collision abort

See the *NET+50/20M Hardware Reference,* Chapter 10, "Ethernet Controller."

The Ethernet transmitter occasionally experiences an Ethernet Transmit Abort due to excessive collisions. A flaw in the Collision Back-Off random number generator causes this problem. The random number generator is not random in all cases. The generator does experience some repetition.

Workaround

None.

MAC module long preamble filtering

See the *NET+50/20M Hardware Reference,* Chapter 10, "Ethernet Controller."

The MAC module receiver has a bug that causes the MAC to automatically filter all Ethernet Receive packets that have a preamble length of 96 bits or greater. When the MAC encounters a packet with a long preamble, the MAC ignores the current packet and begins waiting for the next Ethernet Receive packet. This causes a violation to the 802.3u specification concerning preamble detection.

Workaround

None.

SER module THALF indicator failure writing BYTES/HWORDS

See the *NET+50/20M Hardware Reference*, Chapter 11, "Serial Controller Module."

The THALF bit does not work properly when writing bytes or halfwords to the FIFO data register. The behavior is erratic when writing bytes or halfwords to the FIFO. This can cause an unannounced FIFO overrun condition when polling the THALF bit to determine when more data can be written to the FIFO. This problem does not occur when writing words to the FIFO data register.

The FIFO contains eight rows of 4 byte entries. Any time a word, halfword, or byte is written to the FIFO, one entire row is consumed. The FIFO considers itself half full when 16 bytes have been written to the FIFO, not when four rows have been written. Therefore, writing 4 bytes to the FIFO results in four rows being consumed, with only 4 bytes sitting in the FIFO. The FIFO half condition is triggered only when 16 bytes are in the FIFO, which fails in this condition.

Workaround

Never use the THALF status bit when filling the FIFO using byte or halfword writes. Use only the TEMPTY status bit when writing byte or halfwords.

SYS module HRESET* signal failure

See the schematics that use the PORTC4 connection to reset an external device such as a PHY.

The NET+ARM reset mechanism renders the HRESET* signal unusable because the GPIO port through which HRESET* is guided out of the chip is also reset during an external or hardware reset.

Workaround

Use PORTC4 in GPIO output mode, and reset the PHY (or any other external device) by generating a low-going pulse using the firmware.

SRAM sync burst cycle during DMA

When a chip select is set as a synchronous read with the WAIT set to 0 and the BCYC field set to 00, the chip produces the correct 2-111 burst transfer:

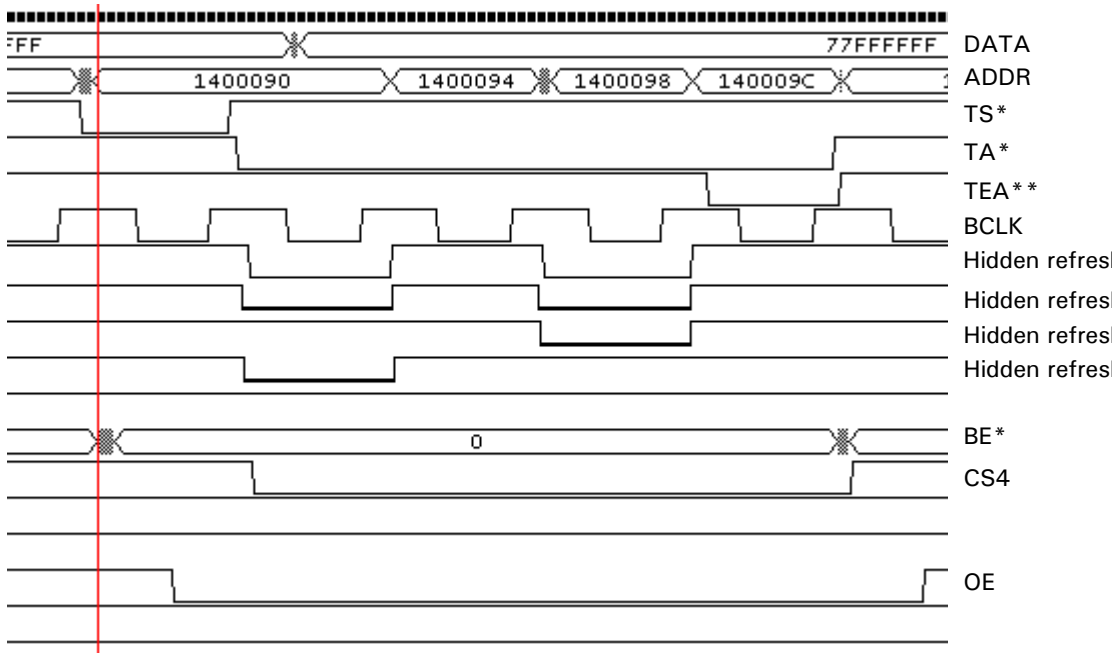


Figure 1: 2-111 burst transfer

An SRAM burst cycle transfer that occurs during a DMA transfer, however, will be a 2-222 burst transfer:

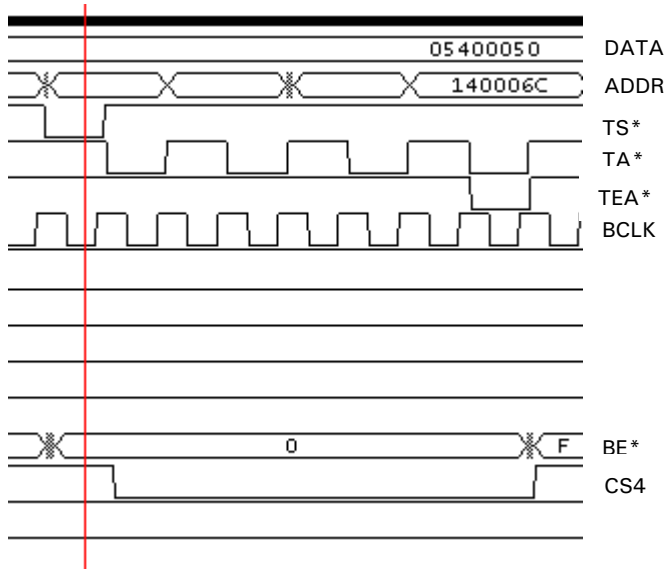


Figure 2: 2-222 burst transfer

Workaround

None.

SDRAM 256 MB mask failure

The NET+50 chip select does not generate the required precharge commands if a value larger than 128 MB is written in the Chip Select Option register mask word. In particular, if the SDRAM is located at 0-256M and one read is done at address 0x400 and another read is done at address 0x0, there is no precharge for the read at address 0.

Workaround

Mask size `0xF0000` (256 MB) should not be used.

Incorrect description of the TPRE bit operation in the *Hardware Reference*

The *NET+50/20M Hardware Reference* states that when the TCLK bit is set to 1, the timer divides SYSCLK by 8 if TPRE is set to 0 and by 4096 if TPRE is set to 1. The SYSCLK is not divided, however; the timer counts clock periods.

Workaround

None needed. You can use the F_{XTAL} input (TCLK = 0) for large timeouts or the SYSCLK input (TCLK = 1) for more resolution.

Erroneous timeouts when loading timer

There can be two erroneous timeouts when the timer is first loaded.

Workaround

In most cases, no workaround is required, because the error averages out over many readings. The problem can become critical if you set the timer to a large value, such as four hours. A workaround for this situation is to program the timer to timeout twice, with a small value such as 4μs, then load the larger value.

Example

```
0xffb00010 = 80000001;
          Wait for the first TIP bit. Reset the TIP bit.
          Wait for the first TIP bit. Reset the TIP bit.
0xffb00010 = 80ffffff;
```

In this way, the two erroneous timeouts will occur with a small value, and the large value can then be set.

Note: Toggling TE (timer enable) to disable and re-enable the timer does not eliminate this issue.

Multiple register store instruction fails to complete

A multiple register store instruction misses the second memory write if interrupted by an Ethernet TxDMA, following a hardware-applied reset.

Workaround

Add a software-applied reset at the beginning of the code.

The *Hardware Reference* does not correctly describe the SIZE field in the DMA Control register (bits 17 and 16 in 0xFF900XXX)

The 2-bit size SIZE field should allow you to configure the data size to be the size of the external device's bus during a fly-by external DMA transfer. Setting the SIZE field should allow you to do read or write fly-by transfers from an 8- or 16-bit device to the NET+50 32-bit DRAM; the data, however, appears in the incorrect byte position, making this feature unusable.

The first table shows the problem when the SIZE field is set to 01 for a 16-bit external bus during a fly-by write transfer. The table shows data from a 16-bit bus (an incrementing pattern from `0123` to `CDEF`) interfacing to the 32-bit system memory bus. The 16 bits that are not connected to the external device are considered *floating high* (`0xFFFF`). This table shows that, on alternate transfers, the floating part of the bus is written.

Data on the bus during a 16-bit fly-by write	Correct transfer	Actual transfer
0x0123FFFF		
0x4567FFFF	0x01234567	0x1234FFFF
0x89ABFFFF		
0xCDEFFFFF	0x89ABCDEF	0x89ABFFFF

The next table shows the problem when the SIZE field is set to 10 for an 8-bit external bus during a fly-by write transfer. The table shows data from an 8-bit bus (an incrementing pattern from 01 to EF) interfacing to the 32-bit system memory bus. The 24 bits that are not connected to the external device are shown as floating high (0xFFFFFF). This table shows that the correct data is written only on every fourth transfer; the floating part of the bus is written on the other three transfers.

Data on the bus during an 8-bit fly-by write	Correct transfer	Actual transfer
0x01FFFFFF		
0x23FFFFFF		
0x45FFFFFF		
0x67FFFFFF	0x01234567	0x01FFFFFF
0x89FFFFFF		
0xABFFFFFF		
0xCDFFFFFF		
0xEFFFFFFF	0x89ABCDEF	0x89FFFFFF

These tables illustrated the transfer only during a fly-by write, but fly-by read transfers are affected also. With a fly-by read transfer, the NET+50 shifts the data across the bus on each transfer rather than placing the data in the most significant 16- or 8-bit positions (as is done with a fly-by write transfer).

Workaround

If the external device's bus is a different size than the NET+50 system bus, you must use a memory-to-memory transfer. You must configure the PS (port size) field in the Chip Select Option register, for the chip select associated with the external device, to match the external device data bus size.

System bus interface pins require external pullup

The *NET+50/20M Hardware Reference* incorrectly describes the several pins in Table 1: NET+50 PQFP/BGA Chip pinout - System bus interface. Replace the appropriate lines in this table with the following information:

Signal	PQFP	BGA	I/O	OD	Description
TS*	164	P5	I/O	8	Add 1K external pullup
TA*	165	R5	I/O	8	Data transfer acknowledge. Add 490-510 Ohm external pullup.
TEA*	166	T4	I/O	8	Transfer error/last acknowledge. Add 490-510 Ohm external pullup.
BR*	167	T5	I/O	4	Add 1K external pullup
BG*	168	R6	I/O	4	
BUSY*	169	P6	I/O	4	Add 1K external pullup.

Workaround

None.

Cache enable bit

In the *NET+50/20M Hardware Reference*, the System Control register is missing the CACHE bit in the bit-9 location. This field applies only to the NET+50.

Workaround

- Set the bit to 0 to disable cache.
- Set the bit to 1 to enable cache.

The CACHE bit must be set to 1 to enable the internal cache memory found in the NET+50 device (this bit has no effect on the NET+20M). If internal cache memory is not being used, set the CACHE bit to 0; this lowers the power consumption of the NET+50. Trying to use the cache or access the cache memory with the CACHE bit set to 0 results in a data abort.

SRAM 3-wait state errata

During memory-to-memory DMA transfers between a static RAM (SRAM) device with its chip select configured to three wait states and SDRAM memory, erratic behavior can occur on the data bus. This can cause undefined instruction and data abort exceptions.

This problem occurs only when the SRAM is configured with three wait states. All other wait-state configurations work correctly.

Workaround

Configure the SRAM's chip select wait state field to any (acceptable) value other than 3.

Receive ENI FIFO erroneously signals that it is always ready to receive data

The DREQI* signal is always asserted low when the EDWRBUFEMP and DMAE* bits are set. This indicates to the external device that the ENI receive FIFO is ready to receive data — even when it is not. The next table shows all settings for the DMAE*, ERX, ERXDMA, and CE fields with the expected (E) behavior of the DREQI* signal and the actual (A) results:

EDWRBUFEMP	DMAE*	ERX	ERXDMA	CE (DMA)	DREQI* (E)	DREQI* (A)
0	X	X	X	X	1	1
X	1	X	X	X	1	1
1	0	0	0	0	1	0
1	0	0	0	1	1	0
1	0	0	1	0	1	0
1	0	0	1	1	1	0
1	0	1	0	0	1	0
1	0	1	0	1	1	0
1	0	1	1	0	1	0
1	0	1	1	1	0	0

The ENI receive FIFO alternatively can be configured to signal availability of space in the receive FIFO in the PINT2 line by setting the DMAE2 bit, with results shown below. (Note that the expected and actual results differ, similar to the [erroneous] results shown in the previous table.)

EDWRBUFEMP	DMAE*	DMAE2	ERX	ERXDMA	CE (DMA)	PINT2 (E)	PINT2 (A)
0	X	X	X	X	X	0	0
X	1	X	X	X	X	0	0
1	0	1	0	0	0	0	1
1	0	1	0	0	1	0	1
1	0	1	0	1	0	0	1
1	0	1	0	1	1	0	1
1	0	1	1	0	0	0	1
1	0	1	1	0	1	0	1
1	0	1	1	1	0	0	1
1	0	1	1	1	1	1	1

Workaround

None. Do not use ENI receive FIFO.

Ethernet transmitter considerations

The *NET+50/20M Hardware Reference* omits the following procedure from Chapter 9, "DMA Controller Module."

Under heavy load, the Ethernet transmitter can, on rare occasions, fail to recover from aborted transmits caused by events such as late collisions and transmit underruns. This situation occurs under the following conditions:

- TXCOLC or TXAUR bits in the Ethernet Transmitter Status register are set to 1.
- Full bit in the current DMA buffer descriptor is set to 1.
- Status field in the DMA buffer descriptor is set to 0.

The transmit operation will not proceed until the condition is corrected.

In general, the problem of transmit underruns can be avoided by running in half duplex rather than full duplex. Late collisions can be eliminated by proper network design; late collisions are caused by too many cascaded levels of hubs, switches, repeaters, and the like.

Correction

To correct an underrun condition, use these steps:

1. Disable the DMA channel by setting the CE bit to 0 in the DMA Control register.
2. Disable the transmit DMA and transmit FIFO by setting the ETXDMA and ETX bits to 0 in the Ethernet General Control register.
3. Initialize the buffer descriptors and DMA buffer descriptor pointer.
4. Enable the transmit DMA and transmit FIFO by setting the ETXDMA and ETX bits to 1 in the Ethernet General Control register.
5. Enable the DMA channel by setting the CE bit to 1 in the DMA Control register.

The transmit DMA now starts again at the beginning. The recovery is implemented in the NET+OS BSP, no data is lost, and there is no other effect on operation.

General Control register diagram errata

The *NET+50/20M Hardware Reference* incorrectly lists the General Control register address, on page 401, as `FF80 0000`. The correct address is `FFA0 0000`.

Workaround

None.

SPI slave mode

SPI slave modes are non-functional. SPI slave logic is unable to sample or drive data on the correct clock edges. This causes byte mismatches.

Workaround

None.

SPI master mode transmit

When the transmit clock is set to inverted mode (`TXCINV = 1`), the SPI master mode transmit cannot drive transmit data changes on the correct transition of the transmit clock, producing shift errors as shown:

TXCINV	CLKINV	Description	What happens
0	0	Clock idles low, data changes falling clock edge (Default)	Operates correctly
0	1	Clock idles high, data changes rising clock edge	Operates correctly
1	0	Clock idles low, data changes rising clock edge	**Shift error** DO NOT USE
1	1	Clock idles high, data changes falling clock edge	**Shift error** DO NOT USE

`TXCINV` = Transmit clock invert

`CLKINV` = Clock invert

Workaround

None. In SPI transmit mode, set `TXCINV = 0` only. See Table 96: Serial Channel Bit-Rate register definition in the *NET+ 50/20M Hardware Reference*.

Error in ENI notation for FIFO Mask/Status register

The ENI notation above the FIFO Mask/Status register illustration (p. 396 in the *NET+50/20M Hardware Reference*) is incorrect.

- The current notation is `ENI => 1 004`.
- The correct notation is `ENI => 1 0004`.

Workaround

None.

Incorrect description of GPIO port resets

The *Net+50/20M Hardware Reference* states incorrectly that the PORTA, PORTB, and PORTC GEN module fields are not reset on software reset.

Workaround

None necessary. The PORTA, PORTB, and PORTC GEN module fields are reset during powerup, *RESET, watchdog conditions, and software reset.

SRAM and FastPage & EDO DRAM timing

The *Net+50/20M Hardware Reference* incorrectly describes the null periods between memory transfers.

Note 1 for Figures 66 through 69 (SRAM Async) should read as follows:

1 There is always at least one null period between memory transfers. There can be more null periods if the next transfer is DMA. Thirteen clock pulses are required for DMA context switching.

Note 1 for Figures 61 through 65 (SRAM Sync) and Figures 71 through 73 (FastPage and EDO) should read as follows:

1 There can be null periods between memory transfers if the next transfer is DMA. Thirteen clock pulses are required for DMA context switching.

Workaround

None.

External peripheral DMA support

The *Net+50/20M Hardware Reference* incorrectly states on page 194:

Note: You can use (turn on) only one DMA channel at a time.

The Net+50 allows the use of one *pair* of channels at a time. The corrected note should read:

Note: Use channels 3 and 4 *or* channels 5 and 6. You cannot use both channel pairs simultaneously.

Workaround

None required.

Corrupt Ethernet receive packets

Under extremely rare conditions, packets passed between the EThernet MAC and the receive FIFO have a packet word transferred twice. The last word of the packet is then dropped. For those Ethernet packets where the higher level protocol does not provide a checksum, corrupted data might be passed to the application.

The problem frequency has been measured at Digi. Tests have shown frequency at one in 20 million for large packets (1518 bytes) and approximately one in 10-500 million for small packets (64 bytes).

Workaround

Using standard protocols such as TCP/IP or UDP with checksum selected, bad packets are discarded by the stack. Retransmission is requested, as appropriate, for the bad packets. Note, however, that implementing the checksum results in latency and performance degradation.

For protocols that do not have built-in error checking, the Ethernet driver provides a software workaround. You can enable an `#ifdef` statement to perform a software checksum. For more information, go to the support website page and look for the application note called *Porting Ethernet Software CRC to NET+OS 5.0, 5.1, and 6.0*.

Transmit buffer closed bit is not functional (SPI only)

The transmit buffer closed (TXBC) bit, D01, in Serial Status Register A (Serial Controller module) does not work as described.

Workaround

To determine when the last character has been transmitted, use the software workaround for the mode you are using:

Interrupt mode:
There are three options in interrupt mode:

1. Waiting
   - If the FIFO was filled using byte writes, wait 9 character times.
   - If the FIFO was filled using word writes, wait 33 character times.
2. Polling TXEMPTY

   Poll TXEMPTY (bit 0) in Serial Channel Status Register A. When the bit indicates empty, allow 1 character time to let the last character exit the shift register.
3. Using TXHALF interrupt

   You can use this method when filling the FIFO using word writes (32-bit transfers).

   Enable the TXHALF interrupt by setting bit 2 in Serial Channel Control Register A. When this interrupt occurs, it indicates that there are no more than 16 bytes remaining in the FIFO. Once it occurs, wait 17 character times — 16 plus 1 character time to allow the last character to exit the shift register.

DMA mode:
Wait 33 character times after receiving the DMA complete interrupt.

Transmit FIFO timing issue

When transmitting characters from the FIFO, you must be sure each character is shifted out before the next character is processed. Otherwise, the first clock and data bit of the new character can be corrupted.

Workaround

Use the software workaround for the mode you are using:

Interrupt mode:
For all but the first character, check the TXEMPTY bit (D00 in the Serial Status Register A).

- If the TXEMPTY bit is not set, write the next character within 1 character time.
- If the TXEMPTY bit is set, wait 1 character time plus 1 bit time before writing the next character.

DMA mode:
You should not have a problem with this issue when working in DMA mode.

External use of TA* and TEA*

Note these issues for TA* and TEA*:

- **TEA*.** Externally generated TEA* burst cycles can cause failures.
- **TA*.** External TA* can be used only in single cycle mode, to terminate chip selects with 8, 16, and 32 bus widths.

Workaround

The TA* signal in the NET+50 must be synchronized with BCLK, and last for one BCLK.

The internal TA synchronizer settings must be as shown:

- Chip Select Option Register B: SYNC — 1 stage (01)
- System Control register — BSYNC: 1 stage (00)

Both TA* and TEA* should have external 490-510 ohm pullups.

Note: TEA* is never driven by the external device.

Design aids

These design aids are available for download from the Digi website:

- A simplified Net+50 BGA pinout diagram.
- A recommended Net+50 minimum configuration guide.

Digi International
11001 Bren Road East
Minnetonka, MN 55343 U.S.A.
United States: +1 877 912-3444
Other locations: +1 952 912-3444
Fax: +1 952 912-4960
www.digi.com/support
www.digi.com

Online problem reporting:
www.digi.com/support/eservice/eservicelogin.jsp