ADSP-2181 Anomaly List for Revisions 0.0-4.0

The list below represents the known anomalies and workarounds for the ADSP-2181. The silicon revision of a particular device can be found on the chip, as shown below:

ADSP-2181	Part Number
-xxx	Speed Grade/Package
xyzzzzzzz-**4.0**	Lot number and **silicon revision**
Dwwyy	Date code

The table below shows which ADSP-2181s have each anomaly. The left hand side of the table lists the type of anomaly and the numbers on the top of the table refer to the particular revision of the silicon. Any box that has an 'x' contained in it has that anomaly. For a complete description of the anomaly, please refer to the subsequent pages.

		0.0	0.1	0.2	1.1	1.2	1.3	2.0	4.0
1.	External Program Execution and Stack Pointers	x	x	x	x	x	x	x	x
2.	BDMA Write Glitch	x	x	x	x	x	x	x	x
3.	State of Memory Mapped Registers After Reset	x	x	x	x	x			
4.	Setting CNTR or OWRCNTR at end of loop	x	x	x	x	x	x		
5.	Type 3 Instruction anomaly	x	x	x	x	x	x		
6.	BDMA and Simultaneous External Operations	x	x	x	x				

Anomaly #1

DSP Stack pointers can be corrupted when executing code from external memory when non-zero wait states are used.

Workarounds:
If external program execution is required, ensure that zero wait-state memory is used.

If non-zero-wait-state memory is used, processor instruction rate should not exceed 33MHz.

If processor speed exceeds 33MHz with non-zero-wait-state memory, ensure that you are operating at the bottom of the PC stack (i.e. not in a subroutine, loop, or interrupt service routine), otherwise anomalous behavior may occur.

Anomaly #2

During a wait-stated Byte DMA write cycle, the DSP will drive data lines D15-D8 (all other data lines are not affected) low for approximately 2ns every processor cycle regardless of the number of BDMA software wait states. This will create a glitch on data lines D15-D8 just prior to the correct data being latched on the bus. The size of the glitch depends on the loading of the data bus. This anomaly effectively changes the formula for the "Data setup before WR~ High" (t_{DW}) timing specification. The formula changes from:

> .5tck-7+w ns   to
> .5tck-7 ns

This anomaly only affects data lines D15-D8 when performing wait stated BDMA writes. The address bus and write strobe work properly with any wait state value.

Workaround:
A latch could be used to capture the information on the BDMA data lines during the period that the information being written is stable. Also, because of the brevity of the glitch, the capacitance on the data bus may be enough to hold the data values to a sufficient level.

Anomaly #3

In some rare cases, certain bits of the memory mapped registers are not initialized properly when reset is activated. Please note this problem is very rare. One easily detectable symptom of this anomaly is a program memory IDMA read cycle which produces incorrect data even though the program memory location was initialized by an IDMA write.

The "random" initialization state can affect the following memory-mapped control registers:

Register Name	Address
sport0 control[*]	3ff6
sport1 control[*]	3ff2
prog flags	3fe6 bits<7:0>

BDMA control	3fe3 bits<15:8>
	3fe3 bits<7:4>
BDMA ext addr	3fe2 bits<13:0>
BDMA int addr	3fe1 bits<13:0>

SPORT control registers do not enable the SPORTs. The SPORTs will remain disabled until specifically enabled by modification of the System Control Register. The System Control Register is not affected by this anomaly.

The following registers may be affected:

> *mstat*
> *pmovlay*
> *dmovlay*

The processor also may not boot correctly. Booting may be affected in the following ways:

BDMA Booting:
BDMA Control, Count, and Address Registers may not be initialized properly, and therefore the ADSP-2183 may not boot in user code correctly.

IDMA Booting:
The DSP will not recognize a PM transfer until a DM access is performed. This first DM transfer will be corrupted.

Workaround:
The basic requirement to workaround the bad state is to do a single internal dm access (which will not read/write the correct data).

A reset following an internal data memory access will cause all registers to come up correctly. No further workaround would then be necessary.

Otherwise, affected registers will need to be corrected for correct 2183 operation.

BDMA booting:
There is no mechanism using the BDMA to do a single DM access. The boot loader will load in 32 words of data. This data may be corrupted, as BDMA Control, External Address, and Internal Address may be set incorrectly. The ADSP-2181 may then execute garbage code. It is likely that this garbage code will execute an internal DM access. A subsequent RESET will return the processor to the proper state. Therefore, allowing the processor to complete its boot cycle, run for several milliseconds, and then perform a second RESET will initialize the processor correctly.

IDMA booting:
An initial single IDMA DM access will allow the IDMA circuit to function normally. The user should include chip initialization code as the first software module in order to put the control and status registers in their correct power-on state. The first instructions to be executed should clear any pending interrupts, reset MSTAT to stop any potential spurious timer activity, and reset all memory-mapped control registers to their proper initial conditions as described in the data sheet and User's Manual.

Anomaly #4

When using the Type 3 instructions CNTR = DM( ) or OWRCNTR = DM( ) as the next-to-last instruction of a counter loop, where the DM( ) contains the value 1,

```
cntr=#;
do SOMETHING until ce;
     instr1;
     instr2;
     ...
     instn;
     owrcntr = DM( ) or cntr = DM( )   /* where DM( ) = 1 */
SOMETHING:          inst;
next_instr;
```

will cause the processor to either loop back to *instr1*, or execute the first instruction outside of the loop, *next_instr*, and then execute *instr2* inside the loop.

Workaround:
Ensure that the CNTR = or OWRCNTR = instruction occurs earlier in the loop, or use a reg=reg instruction to write the CNTR or OWRCNTR.

Anomaly #5

When using the Type 3 instructions ASTAT = DM( ), CNTR = DM( ), or OWRCNTR = DM( ) the cycle prior to the instruction *if not cond rti;* (where cond = CE for CNTR or OWRCNTR instructions; cond = any arithmetic condition for ASTAT instruction), can cause the ADSP-2183 to RTI, or to pop the status (STS) stack.

Workaround:
Insert another instruction between the Type 3 instruction and the conditional instruction, or use a reg = reg instruction to set ASTAT, CNTR, and OWRCNTR.

Anomaly #6

When Byte DMA activity occurs concurrently with an external Program Memory (PM) access and external Data Memory (DM) access with wait states or I/O Memory access with wait states, program execution may terminate until the ADSP-2183 is reset.   Note: All three events, BDMA, external PM access, and wait-stated DM or I/O access, must be present concurrently for this to apply.

Workaround:
Set wait states to 0 for external Data Memory or I/O Memory, or do not enable BDMA transfers when external Data Memory or I/O Memory accesses may occur.