

### Features

- 40 MIPS, 25 ns Instruction Rate.
- CMOS 0,5µm Technology.
- 256-pin BGA Package.
- Low Voltage Supply, 3.0V to 3.6V.
- Memory Address Space - 16 Gbytes.
- Scalar and Vector Data Format:
- 32-bit Scalar Data,
- Vector Data Packed into 64-bit Blocks in the Form of Variable Length Words from 1- up to 64-bits Each.
- 120 MOPS Peak for 32-bit Scalar Data.
- From 40 to 11.500+ MMAC (Million Multiplication and Accumulation per Second) for Vector Data.
- Hardware Vector by Matrix and Matrix by Matrix Multiplication Support.
- Hardware Implemented Programmable Saturation Function.
- Dual Address Generators.
- Registers:
- Eight 32-bit General Purpose Registers,
- Eight 32-bit Address Registers,
- Three Internal 32\*64 bit Memories,
- Control and State Registers.
- 32- and 64-bit VLIW-like Instructions.
- Two 64-bit Programmable External Memory Interfaces.
- Two Byte-width Communication Ports Hardware Compatible with Ports of TMS320C4x.

### General Description

The NeuroMatrix<sup>®</sup> NM6403 is a high performance DSP with combination of VLIW/SIMD architectures. The processor is based on 64-bit NeuroMatrix<sup>®</sup> Core (NMC) that includes two main units: 32/64-bit RISC Core and 64-bit VECTOR co-processor to support vector operations with elements of 1-, 2-, 3- up to 64-bit length. Each of these data types is critical to the standard DSP algorithms, image processing, voice compression and next generation of wireless protocols.

Performance of NM6403 depends of bit length of data and may varies from 40 MMAC (32-bit data) up to 11.5 GMAC (1-bit data). The flexible operand width and ability to scale performance let designers trade off precision and performance to suit their applications.

The presence of two extra wide 64-bit external buses with up to 640 Mbytes/s total throughput and two address generators allows to load operands from one bus, to operate with them and to store results of previous operations on the other bus simultaneously.

The ability to build multiprocessor systems by using two byte-width communication ports, hardware compatible with ports of TI TMS320C4x, and "shared memory" mode on any of external bus permits to successfully exploit NM6403 for large data flow processing in real time.

The NM6403 has been designed by RC "Module" ([www.module.ru](http://www.module.ru)) and produced with use of 0.5µm CMOS technology by Samsung Electronics Corporation. The device is packaged in a 256-pin PBGA and is available with 25 ns instruction cycle time at 3.0 - 3.6 V.

## Table of Contents

1 Architecture Overview .....	4
1.1 Functional Block Diagram and Signal Description .....	4
1.2 RISC Core .....	7
1.2.1 Registers and Arithmetic/Logic Unit .....	8
1.2.2 Data Address Generators .....	8
1.2.3 Program Sequencer .....	9
1.2.4 Control Unit .....	9
1.3 Vector Coprocessor .....	11
1.4 Programmable Memory Interfaces .....	17
1.5 Communication Ports .....	19
1.6 Memory Map .....	21
1.7 Interrupts .....	22
1.8 Instruction Set Summary .....	23
2 Pin Information .....	33
2.1 Pinout and Pin Assignments .....	33
2.2 Input/Output Buffer Description .....	36
2.2.1 Input buffers .....	37
2.2.2 Output buffers .....	37
2.2.3 5V Tolerant I/O Buffers .....	37
3 Electrical Characteristics and Operating Conditions .....	38
3.1 Absolute Maximum Ratings .....	38
3.2 ESD Sensitivity .....	38
3.3 Recommended Operating Conditions .....	39
3.4 DC Electrical Characteristics .....	39
4 Timing Parameters .....	41
4.1 Clock Signals .....	42
4.2 Reset Timing .....	43
4.3 External Interrupt Timing .....	45
4.4 Timer Pin Timing .....	46
4.5 Memory Interface Timing .....	47
4.6 Communication Port Timing .....	50
4.6.1 Communication Port Byte-Transfer Timing (Write and Read) .....	50
4.6.2 Communication Port Word-Transfer Timing .....	50

4.6.3 Communication Token Transfer Sequence from an Input to an Output Port..... 52

4.6.4 Communication Token Transfer Sequence from an Output to an Input Port..... 53

5 Power Dissipation ..... 54

6 Mechanical Data ..... 55

7 Design Considerations..... 56

    7.1 Electrical Design Considerations..... 56

    7.2 Development support ..... 57

    7.3 Product Documentation ..... 57

# 1 Architecture Overview

NeuroMatrix® NM6403 is a high performance DSP with combination of VLIW/SIMD architectures. The processor is based on 64-bit NeuroMatrix® Core (NMC) that includes two main units: 32/64-bit RISC Core and 64-bit VECTOR co-processor to support vector operations with elements of variable bit length (Patent No.2131145 RU). There are two identical programmable interfaces to work with any memory types as well as two communication ports hardware compatible with TI DSP TMS320C4x ports to build multi-processor systems.

## 1.1 Functional Block Diagram and Signal Description

NM6403 is designed for processing of 32-bit scalar data and variable bit length vector data packed into 64-bit data words. The second data type will be named below as packed vector data. The processor block diagram is shown in Figure 1-1. The processor signal description is represented in Table 1.

Processor comprises the following functional units:

- **RISC CORE** - the main functional unit that is designed for arithmetical and logical computations and shift operations over 32-bit data, 32-bit address calculations of data and instructions for memory access and for processor activity control.
- **VCP** - vector coprocessor that performs arithmetical and logical operations under 64-bit packed words of variable bit length vector data.
- **LMI** and **GMI** - two identical programmable external memory interface units, each of which is connected to the external memory with amount of up to  $2^{31}$  32-bit words. Processor supports 32-bit and 64-bit memory access. In the second case two sequential memory words are accessed. Memory addressing is provided by using the same 15-bit address bus in the timesharing mode for page address and address into the page. Page address appears at address bus while new page access only. Each programmable memory interface unit supports up to two memory banks without additional external controller. Memory banks connected to one programmable memory interface may be of various types and may have different sizes and time parameters. Also each programmable memory interface supports three shared memory modes for easy design of multiprocessor systems.
- **CP0** and **CP1** - two identical communication ports, each of which provides point-to-point data transfer via bidirectional byte width link between processor and external device. The ports are designed for design of NM6403 based high-performance multiprocessor systems. Communication ports are compatible with those of TMS320C4x. Each communication port comprises DMA unit that allows 64-bit data transfer between the port and the external memory connected to the global and (or) local buses.

Processor comprises five internal buses for rapid data exchange between the functional units:

- **LOCAL ADDRESS BUS** and **GLOBAL ADDRESS BUS** are used to transfer instruction addresses generated by RISC core and data addresses generated by RISC core in program mode and by communication port in DMA mode to the respective programmable memory interface for memory access.

- **OUTPUT DATA BUS** is used to transfer data that have to be written to external memory from RISC core, vector coprocessor and communication ports to the respective programmable memory interface.
- **INPUT BUS #1** and **INPUT BUS #2** are used to transfer instructions and data fetched from global or local external memory by the respective programmable memory interface unit to any of the main processor functional units. The bus *INPUT BUS #2* is exclusively occupied by scalar data and *INPUT BUS #1* is mainly occupied by vector data. Data transfers in DMA mode and instruction transfers can use any available input bus.

Data move from one scalar register to another and immediate constant load from instruction buffer to scalar register are provided by means of programmable memory interface units using internal buses *OUTPUT DATA BUS* and *INPUT BUS #2*.

Internal buses *INPUT BUS #1*, *INPUT BUS #2* and *OUTPUT DATA BUS* are 64-bit wide. 32-bit or 64-bit transfer operation/cycle via these buses is possible. To increase efficiency of data moving between 32-bit scalar registers and between 32-bit registers and memory these registers are grouped into 64-bit register pairs. Also processor contains a few 64-bit registers. Thus it is possible to say that processor supports 64-bit transfer operations of scalar data.

Instruction fetch is performed by 64-bit words. Each word is a one 64-bit instruction or two 32-bit instructions.

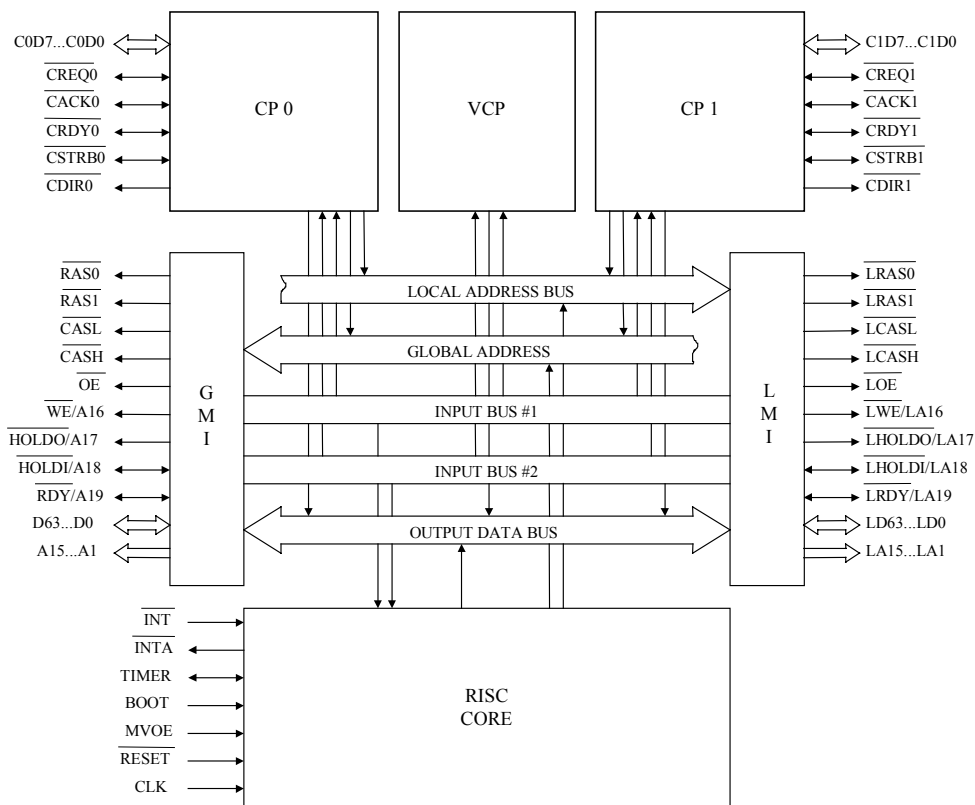


Figure 1-1. Processor Block Diagram.

Table 1-1. Processor Signal Description

Signal <sup>1)</sup>	Pins	Type <sup>2)</sup>	Functional Description
<b>Global(Local) Memory Interface <sup>3)</sup></b>			
(L)D63 – (L)D0	64	I/O	64-bit data bus
(L)A15 – (L)A1	15	O(Z)	15-bit address bus
$\overline{(L)RAS0} / \overline{(L)CS0}^{4)}$	1	O(Z)	Row address strobe for bank 0 of dynamic RAM /page address strobe for bank 0 of static RAM
$\overline{(L)RAS1} / \overline{(L)CS1}^{4)}$	1	O(Z)	Row address strobe for bank 1 of dynamic RAM /page address strobe for bank 1 of static RAM
$\overline{(L)CASL} / \overline{(L)WEL}^{4)}$	1	O(Z)	Column address strobe for low 32-bit of dynamic RAM / write enable for low 32-bit of static RAM
$\overline{(L)CASH} / \overline{(L)WEH}^{4)}$	1	O(Z)	Column address strobe for high 32-bit of dynamic RAM / write enable for high 32-bit of static RAM
$\overline{(L)OE}$	1	O(Z)	Memory output enable
$\overline{(L)WE} / (L)A16^{4)}$	1	O(Z)	Read/write enable for dynamic RAM/16 <sup>th</sup> address bit
$\overline{(L)HOLDO} / (L)A17^{5)}$	1	O(Z)	Internal bus request /17 <sup>th</sup> address bit
$\overline{(L)HOLDI} / (L)A18^{5)}$	1	I/O	External bus request /18 <sup>th</sup> address bit
$\overline{(L)RDY} / (L)A19^{5)}$	1	I/O	Ready /19 <sup>th</sup> address bit
<b>Communication Port CPx (x=0,1) <sup>6)</sup></b>			
CxD7 - CxD0	8	I/O	Data bus
$\overline{CREQx}$	1	I/O	Token request
$\overline{CAACKx}$	1	I/O	Token acknowledge
$\overline{CSTRBx}$	1	I/O	Data strobe
$\overline{CRDYx}$	1	I/O	Ready
$\overline{CDIRx}$	1	O	Transfer direction
<b>Control</b>			
CLK	1	I	Clock
$\overline{RESET}$	1	I	Reset
$\overline{INT}$	1	I	External interrupt
$\overline{INTA}$	1	O	External interrupt acknowledge
BOOTM <sup>7)</sup>	1	I	Boot mode
TIMER <sup>8)</sup>	1	I/O	Timer output

**Notes:**

- 1) A line over a signal name indicates that the signal is active low;
- 2) I - input,  
O - output,  
Z - high impedance.
- 3) The local and global bus interface signals are identical. The local bus interface signal names are the same as global bus interface except the local memory interface signals have an additional prefix «L».
- 4) The functions of signals  $\overline{(L)RAS0} / \overline{(L)CS0}$ ,  $\overline{(L)RAS1} / \overline{(L)CS1}$ ,  $\overline{(L)CASL} / \overline{(L)WEL}$ ,  $\overline{(L)CASH} / \overline{(L)WEH}$  and  $\overline{(L)WE} / (L)A16$  depends on memory type (SRAM or DRAM) used at current memory read/write cycle.
- 5) Signals  $\overline{(L)HOLDO} / (L)A17$ ,  $\overline{(L)HOLDI} / (L)A18$  and  $\overline{(L)RDY} / (L)A19$  are used for shared memory control when processor works at multiprocessor system and are used as additional address bits (from 17<sup>th</sup> to 19<sup>th</sup>) when shared memory not used. At last case the  $\overline{(L)RDY} / (L)A19$  signal also can be used as ready signal from external device during memory access in asynchronous mode.
- 6) The signal functionality of communication port 0 and communication port 1 are identical.
- 7) The BOOTM input must be connected to GND or VDD depending on desired processor boot mode.
- 8) During boot process through global memory (BOOTM=0) TIMER pin is used as input. It specifies memory type for bank 1 of global memory. If system uses SRAM for said memory bank the pull-down should be connected to TIMER pin and the pull-up should be used if DRAM is implemented.

## 1.2 RISC Core

RISC core block diagram is shown in Figure 1-2. It comprises the following functional units:

- **RALU** - register and arithmetic/logic unit;
- **DAG1** and **DAG2** - primary and secondary data address generators;
- **PROGRAM SEQUENCER** - program sequencer;
- **CONTROL UNIT** - control unit.

RISC core comprises the following internal buses that designed for data exchange between RISC core units:

- buses for the first and the second ALU operands;
- bus for the result of arithmetic/logic operation or shift operation;
- buses for the first and the second AU1 operands;
- buses for the first and the second AU2 operands;
- program counter PC input bus.

All internal units and buses of RISC core are 32-bit wide.

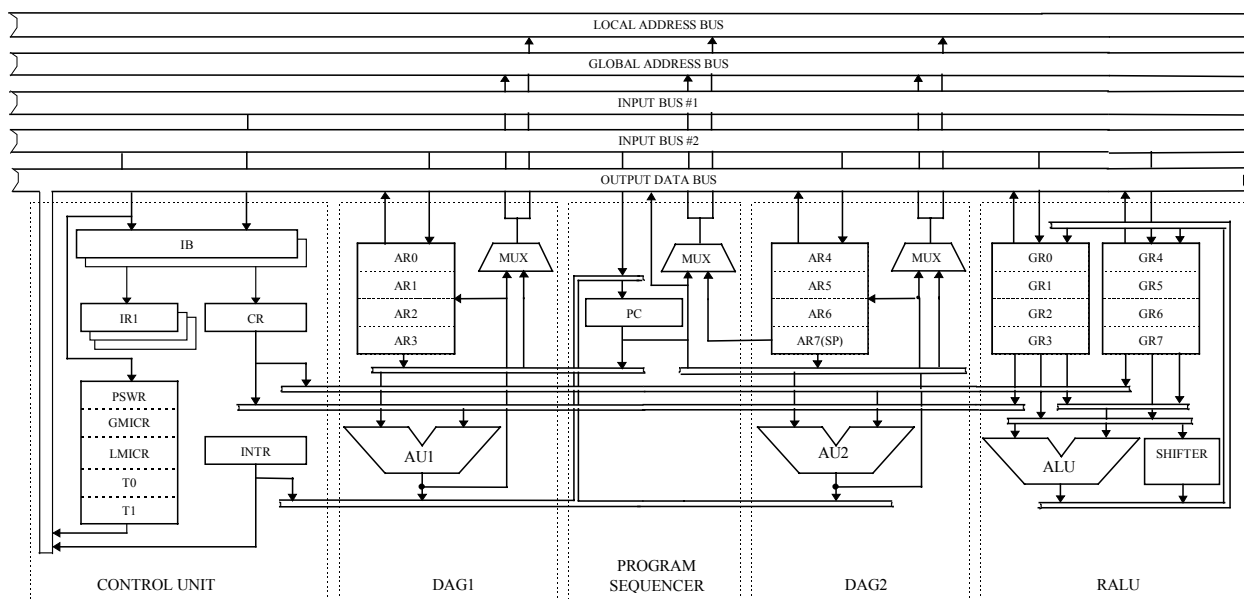


Figure 1-2. RISC core Block Diagram

### 1.2.1 Registers and Arithmetic/Logic Unit

Register and arithmetic/logic unit (RALU) is designed for storage of up to eight 32-bit scalar data and for execution of shift operations and single-operands and double-operands arithmetical and logical operations over them. During execution RALU forms flags: zero result, negative result, carry and arithmetic overflow that can be fixed into program state word register for following reference by conditional branch instructions. Data stored in RALU also can be used as address or address displacement for memory access.

RALU comprises the following subblocks:

- **GR0, ... , GR7** - general purpose registers that are united into a register file. This register file has two input ports respectively connected to the processor *INPUT BUS #2* and to the ALU or shifter result bus and three output ports respectively connected to *OUTPUT DATA BUS* and to the first and the second ALU operand buses. Also the registers GR0, ... , GR3 have an output connected to the second operand bus of AU1. That allows to use data stored in that registers when instruction or data address is generated by DAG1 or when modification of address registers is executed by DAG1. Registers GR4, ... , GR7 have an output connected to the second operand bus of AU2. That allows to use data stored in that registers when instruction or data addresses is generated by DAG2 or when modification of address registers is executed by DAG2.
- **ALU** - arithmetic/logic unit that is able to provide one of sixteen logical or eleven arithmetical operations using data from one or two general purpose registers per one clock cycle. Arithmetical operations are conducted over two's complement 32-bit data.
- **SHIFTER** - barrel shifter unit, that is able to provide left or right arithmetical or logical shift or rotate at any bit count for data placed from any general purpose register to the first ALU operand bus per one clock cycle.

### 1.2.2 Data Address Generators

Two address generators DAG1 and DAG2 are designed for data address generation and instruction address generation for branch instructions. Also these generators provides storage and modification of up to four 32-bit data addresses, branch addresses or address displacements each.

DAG1 comprises the following subblocks:

- **AR0, ... , AR3** - address registers that are united into a register file that has two input ports respectively connected to processor *INPUT BUS #2* and to the AU1 outputs and two output ports respectively connected to processor *OUTPUT DATA BUS* and to the first input operand bus of AU1.
- **AU1** - the first arithmetical unit to perform single operand or double operand arithmetical address calculation or modification of one of DAG1 address registers. Data from AR0, ... , AR3 or from PC can be used as the first operand and data from GR0 - GR3 or 32-bit immediate constant can be used as the second operand.
- **MUX** - address multiplexor to select address from AU1 output or from the first operand bus of AU1 to processor internal address bus depending on the address mode. If MSB address value at

multiplexor output is equal to 0, the address from DAG1 is placed to *LOCAL ADDRESS BUS*, otherwise the address from DAG1 is placed to *GLOBAL ADDRESS BUS*.

The secondary data address generator DAG2 is structurally and functionally very similar to DAG1. The main specific feature of DAG2 is that one of its address registers AR7 also can be used as a stack pointer SP for interrupt processing and subroutine handling instructions.

DAG2 comprises the following subblocks:

- **AR4, ... , AR7(SP)** - address registers that are united into a register file that has two input ports respectively connected to processor *INPUT BUS #2* and to the AU2 output and two output ports respectively connected to processor *OUTPUT DATA BUS* and to the first input operand bus of AU2.
- **AU2** - the second arithmetical unit to perform single operand or double operand arithmetical address calculation or modification of one of DAG2 address registers. Data from AR4, ..., AR7 or from PC can be used as the first operand and data from GR4 - GR7 or 32-bit immediate constant can be used as the second operand.
- **MUX** - address multiplexor to select address from AU2 output or from the first operand bus of AU2 to processor internal address bus depending on address mode. If MSB address value at multiplexor output is equal to 0, address from DAG2 is placed to *LOCAL ADDRESS BUS*, otherwise the address from DAG2 is placed to *GLOBAL ADDRESS BUS*.

### 1.2.3 Program Sequencer

PROGRAM SEQUENCER is intended to generate the address of the next 64-bit instruction or the address of the next pair of 32-bit instructions at linear program blocks, when the next instruction address is defined by an increment operation of the current instruction address. Also PROGRAM SEQUENCER is used to generate the address for access to system stack during call instructions. PROGRAM SEQUENCER comprises the following subblocks:

- **PC** - program counter to contain the address of the next 64-bit instruction or a pair of 32-bit instructions to be fetched. During branch instruction execution the new address feed PC from AU1 or AU2 output. When interrupt is processed, address is taken according register INTR context and when return from subroutine or return from interrupt occurs, address is taken from processor *INPUT BUS #2*. PC output is connected to processor *OUTPUT DATA BUS* that makes it possible to read the value from PC to user program.
- **MUX** - address multiplexor to select address from PC incremented by 2 or from stack pointer AR7(SP) to processor internal address bus. If MSB address value at multiplexor output is equal to 0, the address from PROGRAM SEQUENCER is placed to *LOCAL ADDRESS BUS*, otherwise the address from DAG2 is placed to *GLOBAL ADDRESS BUS*.

### 1.2.4 Control Unit

CONTROL UNIT - performs the initial analysis and decoding of instructions that are fetched from external memory. The unit generates control signals to all processor functional units during instruction execution. It processes interrupt requests and performs request arbitration of the

processor resources such as external and internal buses for other processor functional units. Also it controls external  $\overline{INTA}$  and TIMER signals.

CONTROL UNIT comprises the following subblocks:

- **IB** - instruction buffer that provides store and preliminary analysis of up to four 32-bit or two 64-bit instructions, fetched from external memory.
- **IR1, ... , IR6** - pipeline instruction registers.
- **CR** – register for storage of the immediate constant of a current instruction.
- **GMICR** - global memory interface control register. This register defines global bus configuration, global memory banks boundaries, page size for each global memory bank, operational mode (sync./async.), memory type and time parameters for memory access cycles.
- **LMICR** - local memory interface control register. This register is identical to GMICR, but it performs control under LMI.
- **T0, T1** –programmable timers. They are used to form interrupt signals and signals at external processor output TIMER in a programmable time range. Operation mode for each timer (single or periodical) is specified by a corresponding bits of PSWR register. Also, if either LMICR or GMICR specifies usage of dynamic memory, then the timer T0 is used as DRAM refresh period counter.
- **PSWR** - program state word register. This register is used for storage of the last RALU operation flags, interrupt masks, input/output port initialization modes, input and output channel halt flags of communication ports CP0 and CP1, for control of AFIFO and WFIFO reset, of T0 and T1 timers, of the output TIMER and of the shared memory access. This register is combined with the program counter PC and this register pair is stored at system stack when instruction call or interrupt occurs.
- **INTR** - interrupt and DMA requests register. This register is used for storage of all DMA requests from communication ports CP0 and CP1, and all interrupt requests before they are processed. Additionally this register contains information about AFIFO and WFIFO state (full/empty), communication ports state (receive/transmit), the amount of 64-bit words contained in AFIFO and RAM, and external bus ownership in shared memory mode. Register INTR is connected to processor *OUTPUT DATA BUS* and available for read only.

All registers PSWR, GMICR, LMICR, T0 and T1 are available for user program for read and write. Their inputs are connected to processor *INPUT BUS #2* and outputs are connected to processor *OUTPUT DATA BUS*. Also there is a possibility to set or reset any bit or bits of PSWR by single instruction.

### 1.3 Vector Coprocessor

Vector coprocessor (VCP) is designed to operate with variable bit length data from 1 to 64-bit wide, stored into 64-bit packed data. VCP block diagram is represented in Figure 1-3.

VCP comprises the following functional units:

- **SWITCH 3→2** - switch;
- **SU1, SU2** - saturation units;
- **F1CR, F2CR** - saturation control registers;
- **RCS** - one bit right cyclic shifter;
- **VR** - bias register;
- **RAM** - input operand buffer;
- **SB** - shadow synapse boundary register SB1 and active synapse boundary register SB2;
- **NB** - shadow neuron boundary register NB1 and active neuron boundary register NB2;
- **WEIHGT MATRIXES** - shadow weight matrix and active weight matrix;
- **MU** - multiplication unit;
- **VALU** - vector arithmetic/logic unit;
- **WFIFO** - weight coefficient FIFO;
- **AFIFO** - accumulator FIFO.

Shadow and active weight matrixes, multiplication unit and vector arithmetic/logic unit form operation unit **OU**.

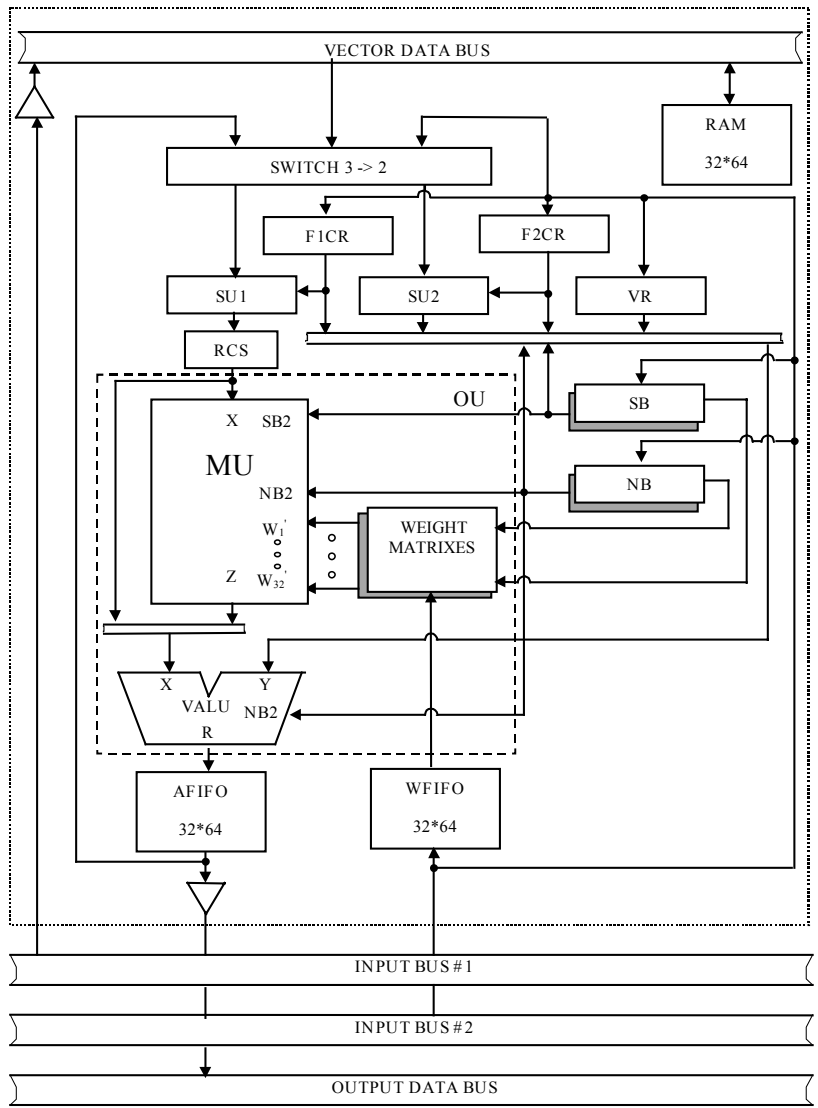


Figure 1-3. Vector Coprocessor (VCP) Block Diagram

**SWITCH 3→2**

SWITCH 3→2 selects two vector operands X and Y from three possible data sources. Depending on the instruction code each vector operand is all zeros operand or is selected from one of the following data sources:

- data from RAM via VECTOR DATA BUS;
- data from AFIFO via feedback bus of VCP;
- data from on-chip/off-chip memory.

## Saturation Units SU1, SU2 and saturation control registers F1CR, F2CR

SU1 and SU2 are used for saturation of 64-bit packed words for both X and Y operands. For each unit SU1 and SU2 there are respective programmable control registers (F1CR for SU1 and F2CR for SU2) whose value defines the number and the bit length of data at packed 64-bit data word and the absolute value of the saturation threshold.

## One Bit Right Cyclic Shifter RCS

Depending on the instruction code, 64-bit data that feed X operand of MU or VALU may be processed by one bit right cyclic shifter RCS. This shift is performed at one clock cycle over packed 64-bit data as a whole.

## Bias Register VR

Bias register VR stores 64-bit bias word. There are some instructions, which can use this register instead of Y operand in multiply-accumulate operation.

## Input Operand Buffer RAM

RAM is a single port memory with the size of 32\*64 bit connected to VCP VECTOR DATA BUS. Input operand buffer RAM is similar to ordinary FIFO with the exception that data at RAM may be read many times.

## Operational Unit OU

OU performs arithmetic and logic operations under 64-bit packed data  $\mathbf{X}=\{X_K \dots X_1\}$ ,  $\mathbf{Y}=\{Y_1 \dots Y_1\}$  and weight coefficient matrix  $W$ , that is read from WFIFO as  $J$  64-bit words of the packed weight coefficients  $W_1=\{W_{11} \dots W_{11}\}, \dots, W_J=\{W_{J1} \dots W_{J1}\}$ . Each operation result is formed as 64-bit packed data word  $R=\{R_1 \dots R_1\}$ . The  $I$  value is to be in the range from 1 up to 64 depending on NB2 register value ( $I$  is equal to the number of ones in NB2), and  $J$  value is to be in the range from 1 up to 32 depending on SB2 register value ( $J$  is equal to the number of ones in SB2).

The  $K$  value depends on the executed vector operation:  $K=I$  - for arithmetic operations,  $K=J$  - for multiply-accumulate operations.

Operation type depends on the instruction code. Operations are executed at pipeline with one operation per cycle throughput rate. OU does not generate any condition flags.

OU performs three types of operations, each of which has own program model as shown in Figure 1-4:

- 16 all possible bit-wise logical operations over 64-bit operands  $X$  and  $Y$ . OU may be presented in this mode as 64 one-bit logical units LU, which concurrently execute the same logical operation over every pair of bits of input operands (see Figure1-4.a). The context of registers SB2 and NB2 is doesn't care for logical operations.
- 4 arithmetical operations over 64-bit packed data  $X$  and  $Y$ :  $X_i + Y_i$ ;  $X_i - Y_i$ ;  $X_i + 1$ ;  $X_i - 1$ , where  $i=1, \dots, I$ . OU may be presented in this mode as  $I$  arithmetic units AU, which concurrently

execute the same arithmetical operation over respective elements of packed data X and Y (see Figure1-4.b). The number and bit length of AU and respective elements of packed operands X, Y and result R are the same and determined by context of register NB2. The context of register SB2 is doesn't care for these arithmetical operations.

- Summation of vector Y with the product of the multiplication of weight coefficient matrix W by vector X (weighted summation):  $R_i = Y_i + \sum_{j=1}^J W_{ij} \times X_j$ , where  $i=1,\dots,I, j=1,\dots,J$ . OU may be

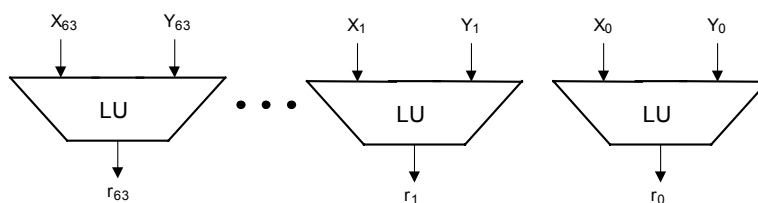
presented in this mode as I concurrently operated circuits, each of them contains J multipliers and one (J+1)-operand adder (see Figure1-4.c). The number I and bit length of respective elements of words Y,  $W_{J,\dots},W_1$  и R are the same and determined by context of register NB2, and the number I and bit length of respective elements of vector X are determined by context of register SB2. Multipliers into OU are implemented on the basis of Booth's algorithm, so the bit length of every element of vector X should be even number.

### Shadow and Active Weight Matrixes

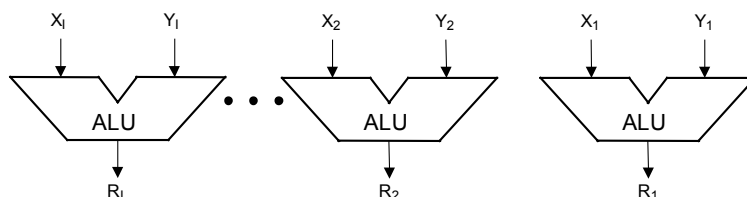
Shadow and active weight matrixes are two memory matrixes with the size of 32\*64 bit each.

Active weight matrix stores weight coefficients that are used in MU operations. Active weight matrix outputs are directly connected to MU inputs and active weight matrix inputs are connected to shadow weight matrix outputs. This architecture allows to load data from shadow weight matrix into active weight matrix per one clock cycle according to LOAD instruction. Simultaneously with the shadow weight matrix to active weight matrix load the vector control registers SB2 and NB2 are loaded from respective SB1 and NB1.

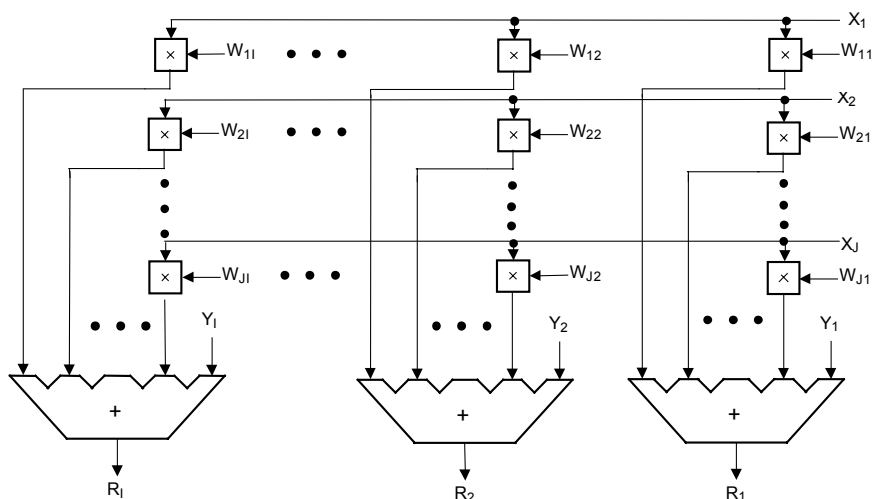
Shadow weight matrix is used to load new weight matrix W from WFIFO while MU operates with the previous weight matrix values stored in active weight matrix. Weight load procedure to shadow weight matrix is initiated by single instruction and processed during 32 clock cycles. The load may be represented as sequential reading of J 64-bit packed weight coefficients  $W_1=\{W_{11} \dots W_{1J}\}, \dots, W_J=\{W_{J1} \dots W_{JJ}\}$  from WFIFO and transforming these coefficients into 32\*64 bit matrix  $W'$ . The J is defined by SB1 value (J is equal to the number of ones in SB1) and number of data and their bit length at packed 64-bit word are defined by NB1. Each i-th row of matrix  $W'$  is a data vector  $W'_k = \{W_{ki} \dots W_{kiJ}\}$ , which will be then multiplied by the k-th pair of bits of vector X ( $k=1,2,\dots,32$ ). All vectors  $W'_1, \dots, W'_{32}$  have the same format as that of any of vectors  $W_1, \dots, W_J$ . This transform allows to implement MU with regular structure that can execute the multiplication of matrix W by vector X per one clock cycle, where elements of matrix or vector may be variable bit length data.



a) Program Model of OU when Logical Operations Are Executed



b) Program Model of OU when Arithmetical Operations Are Executed



c) Program Model of OU when Weighted Summations Are Executed

Figure 1-4. Program Models of OU in Various Modes

### Multiplication Unit MU

MU performs the multiplication of every row  $W_k'$  of matrix  $W'$  by the k-th pair of bits of vector  $X$  ( $k=1,2,\dots,32$ ) on the basis of Booth's algorithm and the summation of obtained partial products. Due to the fact that the matrix  $W'$  is special modification of matrix  $W$ , the result of this operation will be

$Z_i = \sum_{j=1}^J W_{ij} \times X_j$ , where  $i=1,\dots,I$ ,  $j=1,\dots,J$ . The number  $I$  is determined by the context of register

NB2, and the number  $I$  are determined by context of register SB2 (see description of weighed summation execution in OU).

### Vector Arithmetic/Logic Unit VALU

VALU performs arithmetical and logical operations over 64-bit packed data  $X$  and  $Y$  with taking into account the context of register NB2 (see description of execution modes of OU). Depending on the instruction code operand  $X$  may be supplied from RCS output or from MU output .

### Weight Coefficient FIFO WFIFO

Dual port WFIFO has the size of 32\*64 bit and is used as a buffer during weight coefficients loading from external memory to shadow weight matrix. Read and write operations for WFIFO are processed under 64-bit packed weight coefficient words. The internal data bus INPUT BUS #2 is used to load data into WFIFO.

### Accumulator FIFO AFIFO

Dual port AFIFO has the size of 32\*64 bit and is used at VCP as an accumulator for last vector operation result in packed 64-bit word format.

## 1.4 Programmable Memory Interfaces

Processor has two similar programmable external global and local memory interfaces. Each interface represents the following key features:

- Separate 88 pins configuration, each with its own 64-bit data bus, up to 19 bit address bus that allow to output 30-bit address in a multiplexed mode;
- 64 bit data and 32 bit data access support (in the last case LSB of internal 32-bit address is used to address 32-bit data at 64-bit data word);
- one clock cycle for read/write memory access;
- shared memory arbitration signals;
- programmable bank and page mapping with up to two memory banks of different memory type support (SRAM, Flash ROM, DRAM, EDO DRAM);
- page mode support independently for any of memory banks;
- selectable wait states (can be programmed or external ready signal can be used);
- refresh cycles for DRAM and EDO DRAM hardware support (with programmable time parameters);
- interface signals set allows to work with different memory types without external memory controller.

Processor supports both the single processor mode and the multiprocessor mode for any external bus. If external memory is shared between two processors then bus arbitration can be processed without external memory controller.

There are three possible external bus configurations for multiprocessor systems:

- 1) bus configuration type 1 (bank 0 - "common", bank 1 - "common");
- 2) bus configuration type 2 (bank 0 - "own", bank 1 - "common");
- 3) bus configuration type 3 (bank 0 - "own", bank 1 - "not own").

An example of the bus configuration type 1 is shown in Figure 1-5. The key feature of this configuration is that the access to memory banks (MEMORY BANK1, MEMORY BANK2) can be provided by only one processor NP1 or NP2 at the same time.

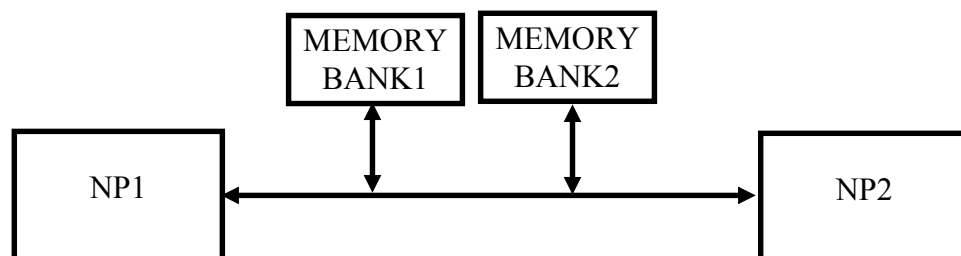


Figure 1-5. Bus Configuration Type 1 (Common Bank 0, Common Bank 1)

An example of the bus configuration type 2 is shown in Figure 1-6. At this configuration each processor has one own memory bank that is inaccessible from other processor (NP1 has own MEMORY BANK1 and NP2 has own MEMORY BANK2) and one memory bank (MEMORY BANK3) is common for both processors. Processors can gain access to this memory bank one by one through external buffers BUFFER1 and BUFFER2.

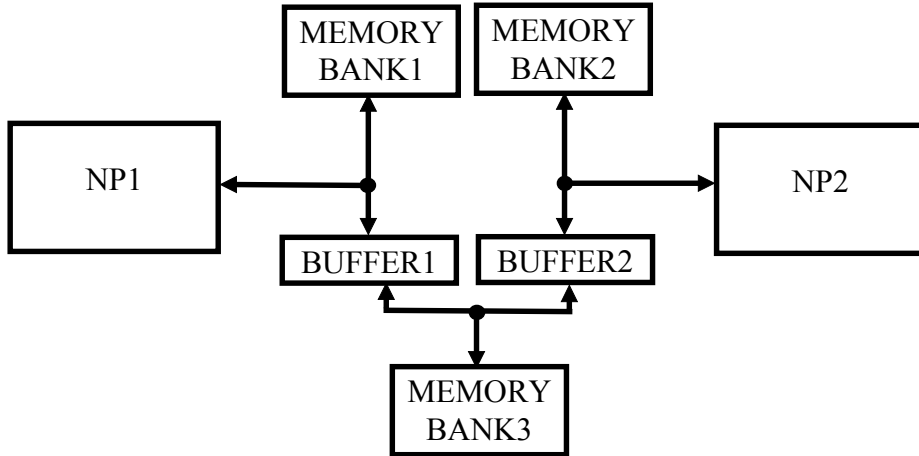


Figure 1-6. Bus Configuration Type 2 (Own Bank 0, Common Bank 1)

An example of bus configuration type 3 is shown in Figure 1-7. At this configuration each processor has one own memory bank (NP1 own MEMORY BANK1 and NP2 own MEMORY BANK2) that is accessible from other processor as second not own memory bank through external BUFFER.

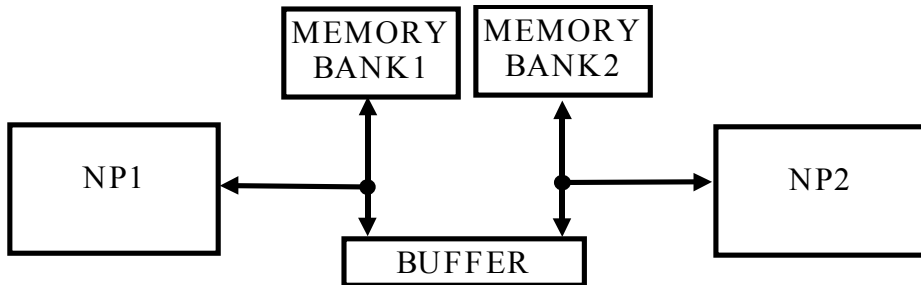


Figure 1-7. Bus Configuration Type 3 (Own Bank 0, not Own Bank 1)

## 1.5 Communication Ports

Processor has two identical high-speed communication ports: port 0 and port 1, each of which provides a point-to-point bi-directional communication interface to another processor or some external peripherals. Figure 1-8 represents an internal architecture of a single communication port.

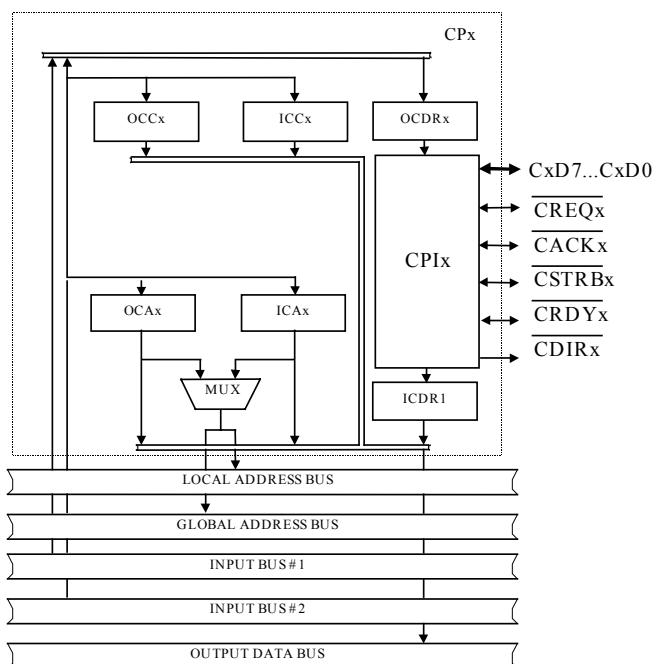


Figure 1-8. Communication Port CP<sub>x</sub> Block Diagram (x = 0, 1)

Each port CP<sub>x</sub> (x = 0,1) contains the following components:

- **CPI<sub>x</sub>** - communication port interface control unit that arbitrates between processor and another device, which of them has possession of the communication data bus at any given time;
- **MUX** - multiplexer for selection of the address source for the communication port DMA request;
- **OCC<sub>x</sub>** - output channel counter that counts the number of 64-bit words during transmission via communication port;
- **OCA<sub>x</sub>** - output channel address register that defines DMA address during transmission via communication port;
- **OCDR** - output channel data register;
- **ICC<sub>x</sub>** - input channel counter that counts the number of 64-bit words during receiving via communication port;
- **ICA<sub>x</sub>** - input channel address register that define DMA address during receiving via communication port;
- **ICDR<sub>x</sub>** - input channel data register.

**Bidirectional data and control lines**

The communication port interface consists of the following bidirectional data and control lines:

- $\overline{CREQ}_x$  - communication port token request;
- $\overline{CACK}_x$  - communication port token acknowledge to relinquish ownership of the communication port data bus upon receiving  $\overline{CREQ}_x$  from another processor;
- $\overline{CSTRB}_x$  - communication port strobe. A sending processor activates this signal to indicate that it has placed a valid data on the communication port data bus;
- $\overline{CRDY}_x$  - communication port ready. A receiving processor activates this signal to indicate that it has received a data byte via the communication port data bus;
- $C_xD(7-0)$  - communication port data bus. This bus carries data bidirectionally between two processors or between a processor and some other device.

There are also output signals  $\overline{CDIR}_x$ , which indicate transfer direction with high level when input and low level when output.

## 1.6 Memory Map

Processor supports 32-bit internal address with 32-bit and 64-bit memory access. The total memory range of the processor is 4Giga 32-bit words or 16Gbyte. This address space is divided into two equal parts: local and global (see Figure 1-9). If address MSB is equal to zero then there is an access to local memory, if address MSB is equal to one then there is an access to global memory. The address LSB is used for access to 32-bit data. If address LSB is equal to zero the data from low memory word (bits 31-0) are used, if address LSB is equal to one the data from high memory word (bits 63-32) are used. While accessing 64-bit data or fetching an instruction the internal address LSB is ignored.

32-bit external memory access is available only for scalar instructions if specified source/destination register is 32-bit wide. If 64-bit register is specified or if the vector instruction is processed the 64-bit data access is used.

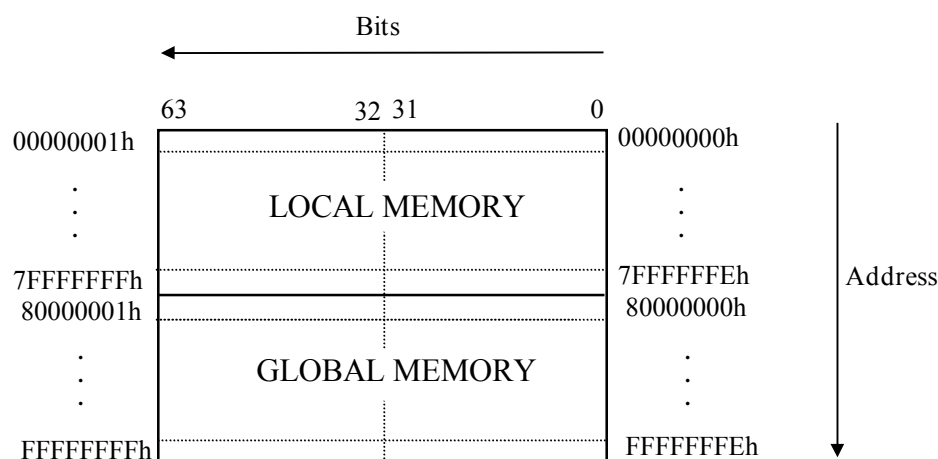


Figure 1-9. Memory Map

## 1.7 Interrupts

Processor supports one external interrupt and 9 internal interrupts:

- two timer interrupts;
- arithmetic overflow during the operation at RISC core interrupt;
- illegal vector instruction interrupt;
- four interrupts concerned with communication port input or output operation finished;
- trace interrupt.

These interrupts are represented in Table 1-2. The priority of interrupts is set according to position in the table - the interrupt with highest priority is at the top and the interrupt with lowest priority is at the bottom of the table. In the case of multiple interrupt request at the same time the interrupt with higher priority will proceed first.

*Table 1-2. Processor Interrupts*

<b>№</b>	<b>Interrupt Source</b>	<b>Interrupt Starting Address</b>
1.	System timer T0	00000000h
2.	Arithmetic overflow during operation at RISC-core	00000008h
3.	Illegal vector instruction	00000010h
4.	External interrupt	00000018h
5.	Input via communication port 1 is finished	00000020h
6.	Input via communication port 0 is finished	00000028h
7.	Output via communication port 1 is finished	00000030h
8.	Output via communication port 0 is finished	00000038h
9.	System timer T1	00000040h
10.	Trace	00000048h

## 1.8 Instruction Set Summary

Processor supports 32-bit and 64-bit VLIW-like instructions (see Figure 1-10). There are four basic instruction types: vector instructions and scalar instructions - control, address register modification and load-store-move with the possibility to set arithmetic operation at the same instruction. All scalar instructions are executed per one clock cycle. Vector instructions may be executed per from 1 up to 32 clock cycles according count field of these instructions.

Load and Store Instructions

Rsrc/ dst  $\leftrightarrow$  (f<sub>adr</sub>.(AR<sub>i</sub>, GR<sub>j</sub>)); AR<sub>i</sub>  $\leftarrow$  f<sub>M</sub>(AR<sub>i</sub>, GR<sub>j</sub>) (R/W = 0 - read; R/W = 1 - write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0	1	AM			R/W	AR <sub>i</sub>		R <sub>src/dst</sub>				SCALAR OP																		

32

Rsrc/ dst  $\leftrightarrow$  (f<sub>adr</sub>.(AR<sub>i</sub>, Displacement)); AR<sub>i</sub>  $\leftarrow$  f<sub>M</sub>(AR<sub>i</sub>, Displacement) (R/W = 0 - read; R/W = 1 - write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	1	0	AMC	R/W	AR <sub>i</sub>		R <sub>src/dst</sub>				SCALAR OP																			
ADDRESS (DISPLACEMENT)																															

32

Register to Register Move Instruction

Rdst  $\leftarrow$  Rsrc

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	1	1	R <sub>dst</sub>				R <sub>src</sub>				SCALAR OP																			

Rdst  $\leftarrow$  constant

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	0	0	R <sub>dst</sub>				0	0	x	x	x	x	SCALAR OP																	
CONSTANT																															

32

Bitwise PSWR set/reset (R/S = 0 - rest; R/S = 1 - set)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	0	0	R/S	1	1	1	0	1	0	0	x	x	x	x	SCALAR OP															
BIT MASK																															

32

Address Register Modification Instructions

AR<sub>j</sub>  $\leftarrow$  f<sub>i</sub>(AR<sub>i</sub>, GR<sub>i</sub>)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	0	1	RM	1	AR <sub>i</sub>		0	1	x	x	AR <sub>j</sub>		SCALAR OP																	

AR<sub>j</sub>  $\leftarrow$  f<sub>i</sub>(AR<sub>i</sub>, Displacement)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	0	0	RMC	1	AR <sub>i</sub>		0	1	x	x	AR <sub>j</sub>		SCALAR OP																	
CONSTANT DISPLACEMENT																															

32

No input/output and address modification operations

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	0	1	xx	0	xxx	0	1	x	x	xx	SCALAR OP																			

No input/output and address modification operations

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	0	0	xx	0	xxx	0	1	x	x	xx	SCALAR OP																			
CONSTANT DISPLACEMENT																															

32

Figure 1-10. Instruction Formats

## Control Instructions

Jump/call to subroutine (J/C = 0 - jump; J/C = 1 - call to subroutine)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0	0	0	PM	J/C	ARi			1	0	Condition			SCALAR OP																	

Jump/call to subroutine with displacement (J/C = 0 - jump; J/C = 1 - call to subroutine)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	1	0	0	PMC	J/C	ARi			1	0	Condition			SCALAR OP																	
63 ADDRESS (DISPLACEMENT) 32																															

Return from subroutine/interrupt (S/I = 0 - return from subroutine; S/I = 1 - return from interrupt)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0	0	0	S/I	0	1	1	1	1	1	1	Condition			SCALAR OP																

## Vector Instructions

AFIFO $\leftrightarrow$ (fadr(ARi, GRi)); ARi $\leftarrow$ f<sub>m</sub>(ARi, GRi) (R/W = 0 - read; R/W = 1 - write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0	0	AM		R/W	ARi			0	1	R_W	W	count			VECTOR OP															

WFIFO $\leftarrow$ (fadr(ARi, GRi)); ARi $\leftarrow$ f<sub>m</sub>(ARi, GRi) (R/W = 0 - read; R/W = 1 - write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0	0	AM		x	ARi			0	0	1	W	count			VECTOR OP															

No input/output operations

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0	0	x	x	x	x	x	x	x	0	0	0	W	x	x	x	x	x	VECTOR OP												

### Legend:

- AM - addressing mode fadr(ARi, GRi) with address register modification ARi $\leftarrow$ f<sub>m</sub>(ARi, GRi) (see Table 1-3).
- AMC- addressing mode using immediate constant fadr.(ARi, Displacement) with address register modification ARi $\leftarrow$ f<sub>m</sub>(ARi, Displacement) (see Table 1-4).
- RM - address register modification mode ARj $\leftarrow$ f<sub>m</sub>(ARi, GRi) (see Table 1-5).
- RMC - address register modification mode using immediate constant ARj $\leftarrow$ f<sub>m</sub>(ARi, Displacement) (see Table 1-6).
- PM - PC modification modes at branch instructions (see Table 1-7).
- PMC - PC modification modes at branch instructions using immediate constant (see Table 1-8).
- ARi - address register i (i = 0, ..., 7).
- ARj - address register j (j = 0, ..., 3 if 24<sup>th</sup> instruction bit is equal to 0, j = 4, ..., 7 if 24<sup>th</sup> instruction bit is equal to 1).
- Condition - condition code used for control instructions (see Table 1-9).
- Rsrc/dst - source/destination register for load, store and move instructions (see Table 1-10).
- Count - vector instruction repeat count, must be in the range from 1 to 32.
- R\_W - RAM write control (0 - no write/1-write).
- W - load weight coefficients from WFIFO to OU control (0 - no load/1 - load).
- SCALAR OP - scalar arithmetic operation code (see Figure 1-11).
- VECTOR OP - vector arithmetic operation code(see Figure 1-12).
- P - parallel instruction flag (0 - disable execution in parallel for current instruction/ 1 - enable execution in parallel for current instruction).

*Figure 1-10. Instruction Formats (Continued)*

## Addressing Modes

Memory addressing modes for 32-bit load and store instructions are defined at AM instruction field (see Table 1-3). Memory addressing modes for 64-bit load and store instructions that use immediate constant are defined at AMC instruction field (see Table 1-4).

Table 1-3. Memory Addressing Modes for 32-bit Load and Store Instructions

AM	$f_{adr.}(AR_i, GR_i)$	$f_m(AR_i, GR_i)$
0 0 0	$GR_i$	$AR_i$
0 0 1	$AR_i+GR_i$	$AR_i+GR_i$
0 1 0	$GR_i$	$GR_i$
0 1 1	Reserved	Reserved
1 0 0	$AR_i$	$AR_i$
1 0 1	$AR_i$	$AR_i+GR_i$
1 1 0	$AR_i-a$	$AR_i-a$
1 1 1	$AR_i$	$AR_i+a$

Table 1-4. Memory Addressing Modes for 64-bit Load and Store Instructions

AMC	$f_{adr.}(AR_i, Displacement)$	$f_m(AR_i, Displacement)$
0 0	Address	$AR_i$
0 1	$AR_i+Displacement$	$AR_i+Displacement$
1 0	Address (Displacement)	Address (Displacement)
1 1	Reserved	Reserved

## Address Register Modification Modes

Address register modification modes for 32-bit scalar instructions are defined by the respective RM instruction field (see Table 1-5). Address register modification modes for 64-bit scalar instructions with immediate constant are defined by the respective RMC instruction field (see Table 1-6).

Table 1-5. Address Register Modification Modes for 32-bit Scalar Instructions

RM	$AR_j \leftarrow f_m(AR_i, GR_i)$
0 0	$AR_i$
0 1	$AR_i+GR_i$
1 0	$GR_i$
1 1	Reserved

Table 1-6. Address register modification modes for 64-bit scalar instructions

RMC	$f_i(AR_i, Displacement)$
0 0	$AR_i$
0 1	$AR_i+Displacement$
1 0	Constant (Displacement)
1 1	Reserved

## PC Modification Modes

PC modification modes for 32-bit branch instructions are defined by the respective PM instruction field (see Table 1-7). PC modification modes for 64-bit branch instructions with immediate constant are defined by the respective PMC instruction field (see Table 1-8).

*Table 1-7. PC Modification Modes for 32-bit Branch Instructions*

PM	PC Modification Modes
0 0	PC:=ARi
0 1	PC:=ARi+GRi
1 0	PC:=GRi
1 1	PC:=PC+GRi

*Table 1-8. PC Modification Modes for 64-bit Branch Instructions*

PMC	PC Modification Modes
0 0	PC:=ARi
0 1	PC:=ARi+Displacement
1 0	PC:=Address (Displacement)
1 1	PC:=PC+ Displacement

## Condition Codes and Flags

Scalar control instructions can be unconditional or conditional depending on Condition instruction field. Decisions of conditional instruction executing are taken depending on the condition codes that are a combination of processor scalar flags: N - negative, Z - zero, V - overflow, C - carry (see Table 1-9).

*Table 1-9. Processor Condition Codes*

Condition	Flag Combination
0 0 0 0	Branch if C=0
0 0 0 1	Branch if V=0
0 0 1 0	Branch if N+Z=0
0 0 1 1	Branch if N=0
0 1 0 0	Branch if (V ⊕ N)+Z=0
0 1 0 1	Branch if V ⊕ N=0
0 1 1 0	Branch if Z=0
0 1 1 1	Unconditional
1 0 0 0	Branch if C=1
1 0 0 1	Branch if V=1
1 0 1 0	Branch if N+Z=1
1 0 1 1	Branch if N=1
1 1 0 0	Branch if (V ⊕ N)+Z=1
1 1 0 1	Branch if V ⊕ N=1
1 1 1 0	Branch if Z=1
1 1 1 1	No Branch

Register Set

Scalar load, store and move instructions field Rsrc/dst defines the register code (see Table 1-10). Depending on the code Rsrc/dst field defines 32-bit registers, 64-bit registers and 64-bit register pairs.

Table 1-10. Load, Store and Move Instructions Field Rsrc/dst

Rsrc/dst	Source Register	Destination Register	Width
0 0 0 0 0 0	AR0	AR0	32
0 0 0 0 0 1	AR1	AR1	32
0 0 0 0 1 0	AR2	AR2	32
0 0 0 0 1 1	AR3	AR3	32
0 0 0 1 0 0	AR4	AR4	32
0 0 0 1 0 1	AR5	AR5	32
0 0 0 1 1 0	AR6	AR6	32
0 0 0 1 1 1	AR7(SP)	AR7(SP)	32
0 0 1 0 0 0	OCA0	OCA0	32
0 0 1 0 0 1	ICA0	ICA0	32
0 0 1 0 1 0	OCA1	OCA1	32
0 0 1 0 1 1	ICA1	ICA1	32
0 0 1 1 0 0	T0	T0	32
0 0 1 1 0 1	LMICR	LMICR	32
0 0 1 1 1 0	GMICR	GMICR	32
0 0 1 1 1 1	PC	PC	32
0 1 0 0 0 0	GR0	GR0	32
0 1 0 0 0 1	GR1	GR1	32
0 1 0 0 1 0	GR2	GR2	32
0 1 0 0 1 1	GR3	GR3	32
0 1 0 1 0 0	GR4	GR4	32
0 1 0 1 0 1	GR5	GR5	32
0 1 0 1 1 0	GR6	GR6	32
0 1 0 1 1 1	GR7	GR7	32
0 1 1 0 0 0	OCC0	OCC0	32
0 1 1 0 0 1	ICC0	ICC0	32
0 1 1 0 1 0	OCC1	OCC1	32
0 1 1 0 1 1	ICC1	ICC1	32
0 1 1 1 0 0	T1	T1	32
0 1 1 1 0 1	--Reserved--	PSWRreset	32
0 1 1 1 1 0	INTR	--Reserved--	32
0 1 1 1 1 1	PSWR	PSWR	32

*Table 1-10. Load, Store and Move Instructions Field Rsrc/dst (Continued)*

<b>R<sub>src/dst</sub></b>	<b>Source Register</b>	<b>Destination Register</b>	<b>Width</b>
1 0 0 0 0 0	GR0, AR0	GR0, AR0	64(32+32)
1 0 0 0 0 1	GR1, AR1	GR1, AR1	64(32+32)
1 0 0 0 1 0	GR2, AR2	GR2, AR2	64(32+32)
1 0 0 0 1 1	GR3, AR3	GR3, AR3	64(32+32)
1 0 0 1 0 0	GR4, AR4	GR4, AR4	64(32+32)
1 0 0 1 0 1	GR5, AR5	GR5, AR5	64(32+32)
1 0 0 1 1 0	GR6, AR6	GR6, AR6	64(32+32)
1 0 0 1 1 1	GR7, AR7	GR7, AR7	64(32+32)
1 0 1 0 0 0	OCC0, OCA0	OCC0, OCA0	64(32+32)
1 0 1 0 0 1	ICC0, ICA0	ICC0, ICA0	64(32+32)
1 0 1 0 1 0	OCC1, OCA1	OCC1, OCA1	64(32+32)
1 0 1 0 1 1	ICC1, ICA1	ICC1, ICA1	64(32+32)
1 0 1 1 0 0	T1, T0	T1, T0	64(32+32)
1 0 1 1 0 1	DIR0	DOR0	64
1 0 1 1 1 0	DIR1	DOR1	64
1 0 1 1 1 1	PSWR, PC	PSWR, PC	64(32+32)
1 1 0 0 0 0	--Reserved--	NBL	32
1 1 0 0 0 1	--Reserved--	SBL	32
1 1 0 0 1 0	--Reserved--	F1CRL	32
1 1 0 0 1 1	--Reserved--	F2CRL	32
1 1 0 1 0 0	--Reserved--	NBH	32
1 1 0 1 0 1	--Reserved--	SBH	32
1 1 0 1 1 0	--Reserved--	F1CRH	32
1 1 0 1 1 1	--Reserved--	F2CRH	32
1 1 1 0 0 0	--Reserved--	NB	64
1 1 1 0 0 1	--Reserved--	SB	64
1 1 1 0 1 0	--Reserved--	F1CR	64
1 1 1 0 1 1	--Reserved--	F2CR	64
1 1 1 1 0 0	--Reserved--	VR	64
1 1 1 1 0 1	--Reserved--	PSWRset	32
1 1 1 1 1 0	--Reserved--	VRL	32
1 1 1 1 1 1	--Reserved--	VRH	32

### Scalar Arithmetic and Logic Operations

In scalar instruction it is possible to set an arithmetic, logic or shift operation over general purpose registers GR0 - GR7 in the SCALAR OP instruction field (see Figure 1-10). Instruction field format is shown in Figure 1-11.

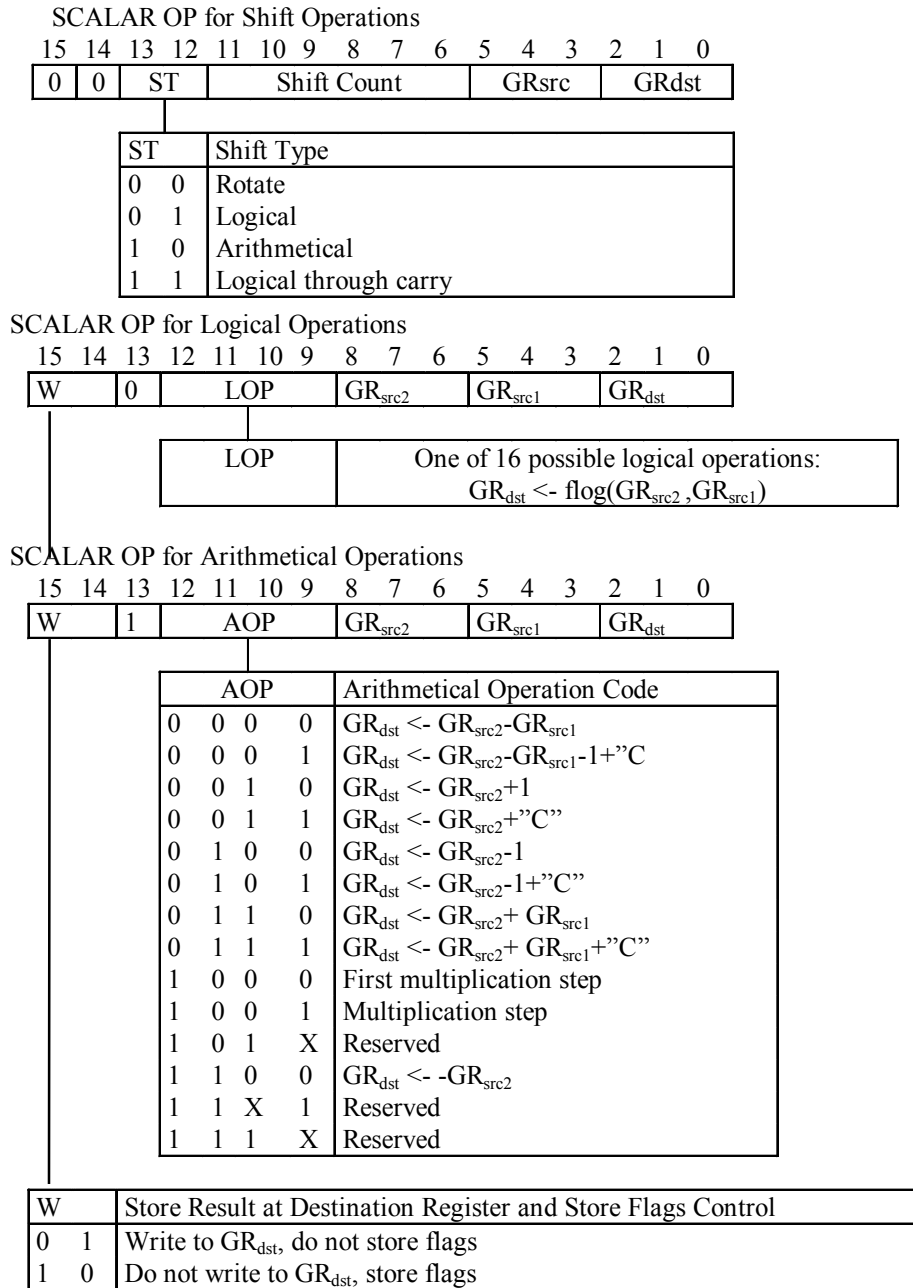


Figure 1-11. SCALAR OP Instruction Field Format

## Vector Arithmetic and Logic Operations

In vector instruction it is possible to set an arithmetic or logic operation over vectors of packed data by VECTOR OP instruction field (see Figure 1-10). VECTOR OP instruction field format is shown in Figure 1-12.

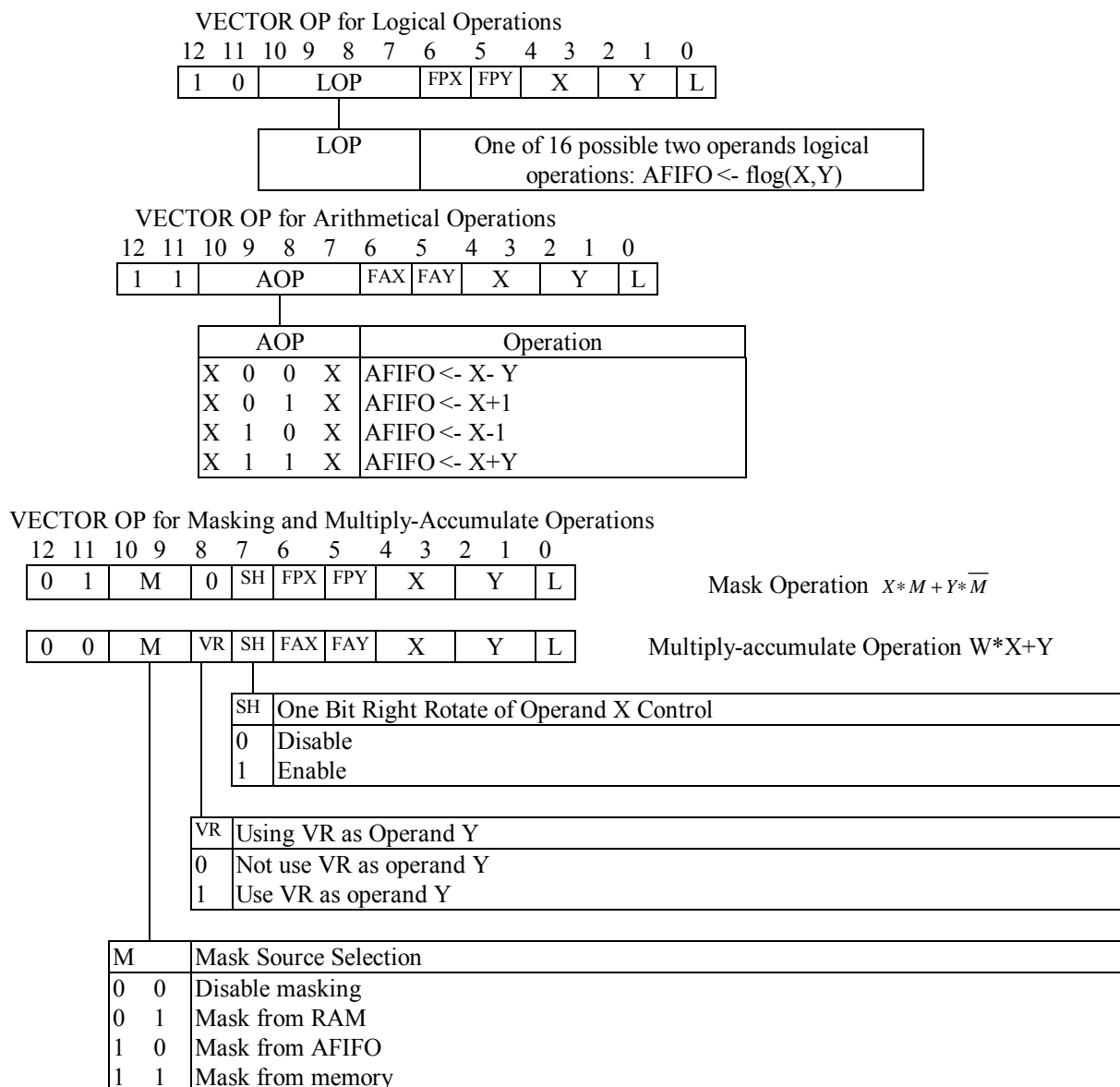


Figure 1-12. VECTOR OP Instruction Field Format

VECTOR OP for Vector Control Registers F2CR, F1CR, NB2, SB2,  
VR write at AFIFO Operation

0	1	00	1	X	X	X	00	00	L
---	---	----	---	---	---	---	----	----	---

VECTOR OP for No Operation

0	0	00	0	X	X	X	00	00	L
---	---	----	---	---	---	---	----	----	---

**Legend:**

X and Y - operands:

00 - zero operand;

01 - operand is taken from RAM;

10 - operand is taken from AFIFO;

11 - operand is taken from external memory.

FPX and FPY - threshold control for X and Y operands:

0 - disable;

1 - enable.

FAX and FAY - saturation control for X and Y operands:

0 - disable;

1 - enable.

L(LOAD) - reload weight coefficients from shadow weight matrix WBUF to operational weight matrix WOPER:

0 - disable;

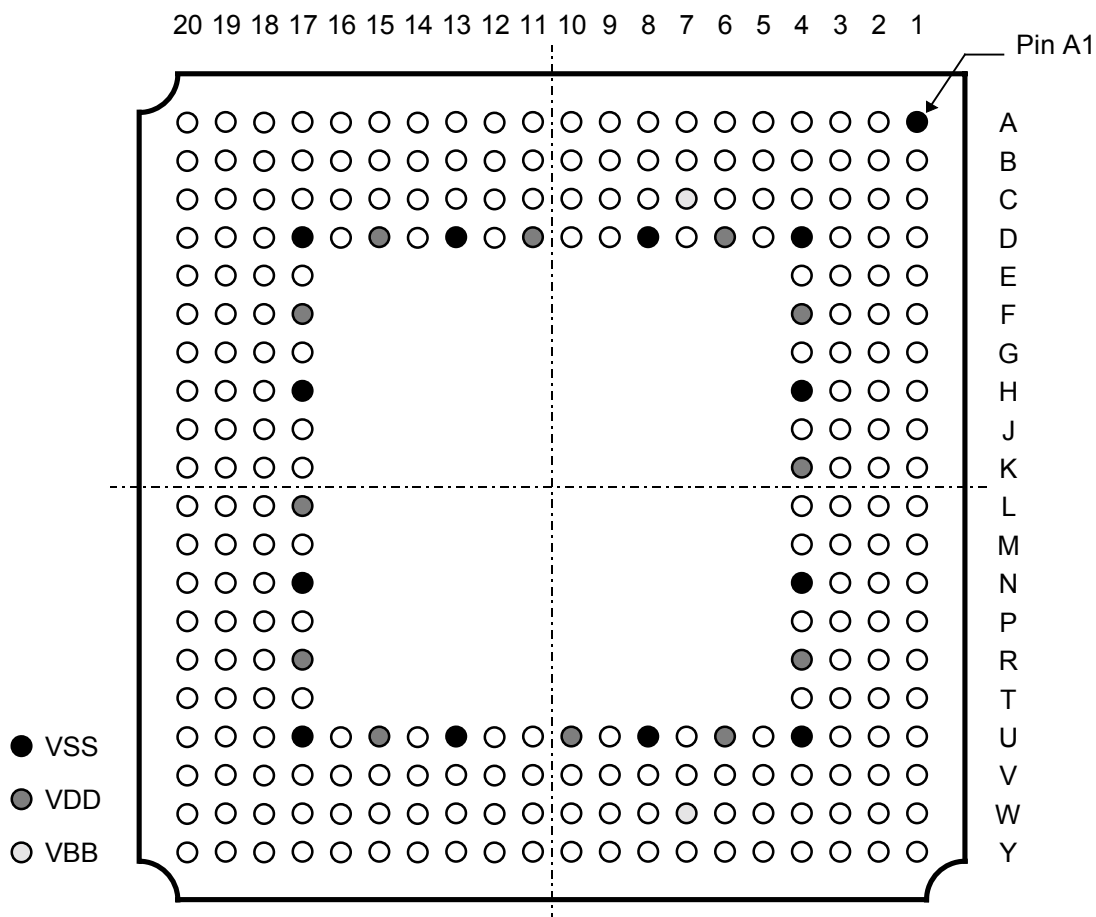
1 - enable.

*Figure 1-12. VECTOR OP Instruction Field (Continued)*

## 2 Pin Information

### 2.1 Pinout and Pin Assignments

The NM6403 is available in a 256-pin plastic ball grid array (PBGA) package. The pinout of this package is shown in Figure 2-1.



*Figure 2-1. NM6403 Pinout (Bottom View)*

Pin assignments are listed in the following tables:

- Table 2-1: Pins sorted by signal name (alphanumeric listing).
- Table 2-2: Pins sorted by pin number (location on Figure 2-1).

Functional descriptions of pin are found in on page 6.

NC (no connection) is the reserved pin, which must be left open and unconnected.

Table 2-1. Pin Assignments Sorted by Signal Name

Signal	Pin N°	Signal	Pin N°	Signal	Pin N°	Signal	Pin N°	Signal	Pin N°
A1	D10	D4	A20	D60	V3	LD25	U20	NC	A16
A2	C10	D5	C18	D61	Y1	LD26	T17	NC	A18
A3	B10	D6	C19	D62	Y2	LD27	U19	NC	B6
A4	A10	D7	E17	D63	V4	LD28	W20	NC	B14
A5	C11	D8	E18	HOLDI/A18	D14	LD29	W19	NC	D19
A6	B11	D9	E19	HOLDO/A17	B15	LD30	Y19	NC	F3
A7	A12	D10	G17	INT	C9	LD31	V17	NC	J19
A8	B12	D11	F19	INTA	D9	LD32	C4	NC	L1
A9	C12	D12	F20	LA1	W10	LD33	B2	NC	P19
A10	D12	D13	G20	LA2	V10	LD34	C3	NC	T2
A11	A13	D14	H19	LA3	Y10	LD35	C2	NC	U16
A12	B13	D15	J17	LA4	Y11	LD36	D3	NC	V5
A13	C13	D16	J20	LA5	W11	LD37	C1	NC	V8
A14	A14	D17	K18	LA6	U11	LD38	E3	NC	V11
A15	C14	D18	K20	LA7	Y12	LD39	E1	NC	Y7
BOOTM	B9	D19	L18	LA8	W12	LD40	F2	NC	Y15
C0D0	Y6	D20	M20	LA9	V12	LD41	G3	OE	R18
C0D1	V7	D21	M18	LA10	U12	LD42	G1	RAS0/CS0	R20
C0D2	W8	D22	N20	LA11	Y13	LD43	H2	RAS1/CS1	P18
C0D3	Y8	D23	N18	LA12	W13	LD44	J4	RDY/A19	C15
C0D4	U9	D24	T20	LA13	V13	LD45	J2	RESET	R3
C0D5	V9	D25	T18	LA14	Y14	LD46	K2	TEST_OUT 1	Y3
C0D6	W9	D26	V20	LA15	W14	LD47	K1	TEST_OUT 2	Y4
C0D7	Y9	D27	U18	LCASH/LWEH	V16	LD48	L3	TIMER	P4
C1D0	D7	D28	V19	LCASL/LWEL	Y17	LD49	M1	VBB	C7
C1D1	A5	D29	Y20	LD0	C16	LD50	M3	VBB	W7
C1D2	A6	D30	V18	LD1	D16	LD51	N1	VDD	D6
C1D3	B7	D31	W18	LD2	B17	LD52	N3	VDD	D11
C1D4	A7	D32	D5	LD3	A19	LD53	P2	VDD	D15
C1D5	C8	D33	B3	LD4	B19	LD54	P3	VDD	F4
C1D6	B8	D34	A2	LD5	B20	LD55	T1	VDD	F17
C1D7	A8	D35	B1	LD6	D18	LD56	T3	VDD	K4
CACK0	U7	D36	D2	LD7	C20	LD57	V1	VDD	L17
CACK1	B5	D37	E4	LD8	D20	LD58	U3	VDD	R4
CASH/WEH	R19	D38	D1	LD9	F18	LD59	W1	VDD	R17
CASL/WEL	P17	D39	E2	LD10	E20	LD60	W2	VDD	U6
CDIR0	W5	D40	G4	LD11	G18	LD61	W3	VDD	U10
CDIR1	B4	D41	F1	LD12	G19	LD62	W4	VDD	U15
CLK	V15	D42	G2	LD13	H18	LD63	U5	VSS	A1
CRDY0	Y5	D43	H3	LD14	H20	LHOLDI/LA18	Y16	VSS	D4
CRDY1	C5	D44	H1	LD15	J18	LHOLDO/LA17	W15	VSS	D8
CREQ0	W6	D45	J3	LD16	K17	LOE	W16	VSS	D13
CREQ1	C6	D46	J1	LD17	K19	LRAS0/LCS0	W17	VSS	D17
CSTRB0	V6	D47	K3	LD18	L20	LRAS1/LCS1	Y18	VSS	H4
CSTRB1	A4	D48	L2	LD19	L19	LRDY/LA19	U14	VSS	H17
D0	B16	D49	L4	LD20	M19	LWE/LA16	V14	VSS	N4
D1	A17	D50	M2	LD21	M17	MVOE	A9	VSS	N17
D2	C17	D51	M4	LD22	N19	NC	A3	VSS	U4
D3	B18	D52	N2	LD23	P20	NC	A11	VSS	U8
		D53	P1	LD24	T19			VSS	U13
		D54	R1					VSS	U17
		D55	R2					WE/A16	A15
		D56	U1						
		D57	U2						
		D58	T4						
		D59	V2						

*Table 2-2. Pin Assignments Sorted by Pin Number*

Pin N°	Signal	Pin N°	Signal	Pin N°	Signal	Pin N°	Signal	Pin N°	Signal
A1	VSS	C14	A15	H17	VSS	R3	RESET	V17	LD31
A2	D34	C15	RDY /A19	H18	LD13	R4	VDD	V18	D30
A3	NC	C16	LD0	H19	D14	R17	VDD	V19	D28
A4	CSTRB1	C17	D2	H20	LD14	R18	OE	V20	D26
A5	C1D1	C18	D5	J1	D46	R19	CASH	W1	LD59
A6	C1D2	C19	D6	J2	LD45		/WEH	W2	LD60
A7	C1D4	C20	LD7	J3	D45	R20	RAS0 /	W3	LD61
A8	C1D7	D1	D38	J4	LD44		CS0	W4	LD62
A9	MVOE	D2	D36	J17	D15	T1	LD55	W5	CDIR0
A10	A4	D3	LD36	J18	LD15	T2	NC	W6	CREQ0
A11	NC	D4	VSS	J19	NC	T3	LD56	W7	VBB
A12	A7	D5	D32	J20	D16	T4	D58	W8	C0D2
A13	A11	D6	VDD	K1	LD47	T17	LD26	W9	C0D6
A14	A14	D7	C1D0	K2	LD46	T18	D25	W10	LA1
A15	WE /A16	D8	VSS	K3	D47	T19	LD24	W11	LA5
A16	NC	D9	INTA	K4	VDD	T20	D24	W12	LA8
A17	D1	D10	A1	K17	LD16	U1	D56	W13	LA12
A18	NC	D11	VDD	K18	D17	U2	D57	W14	LA15
A19	LD3	D12	A10	K19	LD17	U3	LD58	W15	LHOLD0
A20	D4	D13	VSS	K20	D18	U4	VSS		/LA17
B1	D35	D14	HOLDI/A18	L1	NC	U5	LD63	W16	LOE
B2	LD33	D15	VDD	L2	D48	U6	VDD	W17	LRAS0
B3	D33	D16	LD1	L3	LD48	U7	CACK0		/LCS0
B4	CDIR1	D17	VSS	L4	D49	U8	VSS	W18	D31
B5	CACK1	D18	LD6	L17	VDD	U9	C0D4	W19	LD29
B6	NC	D19	NC	L18	D19	U10	VDD	W20	LD28
B7	C1D3	D20	LD8	L19	LD19	U11	LA6	Y1	D61
B8	C1D6	E1	LD39	L20	LD18	U12	LA10	Y2	D62
B9	BOOTM	E2	D39	M1	LD49	U13	VSS	Y3	TEST_OUT
B10	A3	E3	LD38	M2	D50	U14	LRDY /LA1		1
B11	A6	E4	D37	M3	LD50		9	Y4	TEST_OUT
B12	A8	E17	D7	M4	D51	U15	VDD		2
B13	A12	E18	D8	M17	LD21	U16	NC	Y5	CRDY0
B14	NC	E19	D9	M18	D21	U17	VSS	Y6	C0D0
B15	HOLD0	E20	LD10	M19	LD20	U18	D27	Y7	NC
B16	/A17	F1	D41	M20	D20	U19	LD27	Y8	C0D3
B17	D0	F2	LD40	N1	LD51	U20	LD25	Y9	C0D7
B18	LD2	F3	NC	N2	D52	V1	LD57	Y10	LA3
B19	D3	F4	VDD	N3	LD52	V2	D59	Y11	LA4
B20	LD4	F17	VDD	N4	VSS	V3	D60	Y12	LA7
C1	LD37	F18	LD9	N17	VSS	V4	D63	Y13	LA11
C2	LD35	F19	D11	N18	D23	V5	NC	Y14	LA14
C3	LD34	F20	D12	N19	LD22	V6	CSTRB0	Y15	NC
C4	LD32	G1	LD42	N20	D22	V7	C0D1	Y16	LHOLDI
C5	CRDY1	G2	D42	P1	D53	V8	NC		/LA18
C6	CREQ1	G3	LD41	P2	LD53	V9	C0D5	Y17	LCASL
C7	VBB	G4	D40	P3	LD54	V10	LA2		/LWEL
C8	C1D5	G17	D10	P4	TIMER	V11	NC	Y18	LRAS1
C9	INT	G18	LD11	P17	CASL	V12	LA9		/LCS1
C10	A2	G19	LD12		/WEL	V13	LA13	Y19	LD30
C11	A5	G20	D13	P18	RAS1/CS1	V14	LWE/LA16	Y20	D29
C12	A9	H1	D44	P19	NC	V15	CLK		
C13	A13	H2	LD43	P20	LD23	V16	LCASH		
		H3	D43	R1	D54		/LWEH		
		H4	VSS	R2	D55				

## 2.2 Input/Output Buffer Description

There are 12 different types of I/O buffers. This section briefly describes

The I/O structure has been fully characterized for ESD protection and latch-up resistance.

Table 2-3. Input/Output Buffer

Signal	Pins	Type	Schmitt trigger	Pull-up	Pull-down	5 V Tolerant	Drive [mA]	Slew rate
<b>Global (Local) Bus External Interface</b>								
(L)D63 - (L)D0	64	I/O					4	+
(L)A15 - (L)A1	15	O(Z)					8	+
(L)RAS0 / (L)CS0	1	O(Z)					8	
(L)RAS1 / (L)CS1	1	O(Z)					8	
(L)CASL / (L)WEL	1	O(Z)					8	
(L)CASH / (L)WEH	1	O(Z)					8	
(L)OE	1	O(Z)					12	
(L)WE / (L)A16	1	O(Z)					8	+
(L)HOLD0 / (L)A17	1	O(Z)					8	+
(L)HOLD1 / (L)A18	1	I/O					8	+
(L)RDY / (L)A19	1	I/O		+			8	+
<b>Communication Ports #x Interface (x = 0, 1)</b>								
CxD7 - CxD0	8	I/O	+			+	6	
CREQx	1	I/O	+	+		+	6	
CACKx	1	I/O	+	+		+	6	
CSTRBx	1	I/O	+	+		+	6	
CRDYx	1	I/O	+	+		+	6	
CDIRx	1	O					4	
<b>General Control</b>								
CLK	1	I						
RESET	1	I						
TIMER	1	I/O					4	
INT	1	I						
INTA	1	O					4	
BOOTM	1	I						
MVOE	1	I			+			
TEST_OUT1	1	O					2	
TEST_OUT2	1	O					2	

There are 3 functional types of I/O buffers: input, output and bi-directional buffers.

Each bi-directional buffer is combination some input buffer and some tri-state output buffer in a single unit.

### 2.2.1 Input buffers

There are two basic types of input receivers in NM6403: CMOS input buffer and Schmitt trigger input buffer. Schmitt trigger input buffers is better than CMOS input buffer against a noise because each Schmitt trigger input buffer has two different input thresholds for positive- and negative-going signals. This hysteresis between positive- and negative-going voltage signals can filter a noisy signal to a wanted one.

10 input buffers consist 100K $\Omega$  pull-up resistors. Only one input buffer (MVOE) has 100K $\Omega$  pull-down resistor.

### 2.2.2 Output buffers

There are two basic types of output buffers: non-inverting and tri-state (with higher impedance state).

The output buffers are available in a range of driving capabilities from 2 mA to 12 mA. Current level of output buffer affects their propagation delay and current noise.

The slew rate control is provided for some buffers to reduce output power/ground bus noise and signal ringing.

### 2.2.3 5V Tolerant I/O Buffers

NM6403 was designed using 0.5  $\mu\text{m}$  CMOS process, which has the most optimum performance in 3.3V. In this process, voltages more than 3.6V are not allowed at the gate oxide because of a reliability problem. A special circuit is adopted in order to make pin voltage tolerable up to 5.25 V. Obviously, this circuit is constructed not to permit more than 3.6 V at the gate oxide.

The output drive capability of tolerant I/O buffers is up to 6 mA.

The maximum external tolerance voltage of this buffer is 5.5V, and the leakage current of tri-state tolerant pin is less than 10 nA in 0 – 5 V and less than 70  $\mu\text{A}$  in 5.5 V.

When the tolerant I/O buffer is tri-state and the output pin voltage is 5 V, 5 V of bulk bias voltage is required to prevent current from flowing through a chip, however, almost no current flows.

If the bulk bias is 3.3 V, it operates as a 3.3 V normal buffer.

### 3 Electrical Characteristics and Operating Conditions

The following electrical characteristics and requirements are preliminary information and are subject to change.

#### 3.1 Absolute Maximum Ratings

Stresses beyond those listed in the Table 3-1 may cause permanent damage to the device. These are stress ratings only; functional operation of the device at these or any other conditions above those indicated in the Table 3-2 (Recommended Operating Conditions) on page 38 is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Table 3-1. Absolute Maximum Ratings

Symbol	Rating <sup>1,2</sup>	Value	Unit <sup>3</sup>	
V <sub>DD</sub>	Supply voltage	-0.3 to +4.6	V	
V <sub>BB</sub>	PMOS bulk bias	-0.3 to +6.0	V	
V <sub>IN</sub>	Input voltage	tolerant inputs <sup>4</sup>	-0.3 to V <sub>BB</sub> +0.3	V
		all other inputs	-0.3 to V <sub>DD</sub> +0.3	V
V <sub>OUT</sub>	Output voltage swing	-0.3 to V <sub>DD</sub> +0.3	V	
I <sub>IN</sub>	Input current	-10 to +10	mA	
T <sub>A</sub>	Operating free-air temperature (Ambient)	-40 to +85	°C	
T <sub>STG</sub>	Storage temperature	-40 to +125	°C	

- Notes:**
1. All input ratings are valid for both inputs only and bidirectional pins.
  2. All output ratings are valid for both outputs only and bidirectional pins.
  3. All voltage values are with respect to V<sub>SS</sub>.
  4. Tolerant bidirectional pins: C0D7-C0D0,  $\overline{\text{CREQ0}}$ ,  $\overline{\text{CACK0}}$ ,  $\overline{\text{CSTRB0}}$ ,  $\overline{\text{CRDY0}}$ , C1D7-C1D0,  $\overline{\text{CREQ1}}$ ,  $\overline{\text{CACK1}}$ ,  $\overline{\text{CSTRB1}}$ ,  $\overline{\text{CRDY1}}$ .

#### 3.2 ESD Sensitivity

The NM6403 processor is an ESD (Electrostatic discharge) sensitive device. Electrostatic charges readily accumulate on the human body and equipment and can discharge without detection. Permanent damage may occur to devices subjected to high energy electrostatic discharges.

The NM6403 processor includes ESD protection circuitry to dissipate high energy discharges. RC “Module” employs a human-body model for ESD-susceptibility testing. Since the failure voltage of electronic devices is dependent on the current, voltage, and hence, the resistance and capacitance, it is important that standard values be employed to establish a reference by which to compare test data. Values of 100 pF and 1500 Ω are the most common and are the values used in the human-body model test circuit. The breakdown voltage for the NM6403 is greater than 2000 V.

Although input protection circuitry has been incorporated into the devices, proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Unused devices must be stored in conductive foam or shunts, and the foam should be discharged to the destination socket before devices are removed.

### 3.3 Recommended Operating Conditions

Electrical and thermal requirements refer to operating conditions imposed on the user for proper operation of the device.

Table 3-2. Recommended Operating Conditions

Symbol	Parameter <sup>1,2</sup>	Min	Nom <sup>3</sup>	Max	Unit <sup>4</sup>
V <sub>DD</sub>	Supply voltage	3.0	3.3	3.6	V
V <sub>BB5</sub>	PMOS 5V bulk bias	4.75	5.0	5.25	V
V <sub>BB3</sub>	PMOS 3V bulk bias	3.0	3.3	3.6	V
V <sub>SS</sub>	Supply voltage	0			V
V <sub>IH</sub>	High-level input voltage	tolerant inputs <sup>5</sup>	2.1	V <sub>BB</sub> +0.3	V
		all other inputs	0.7V <sub>DD</sub>	V <sub>DD</sub> +0.3	V
V <sub>IL</sub>	Low-level input voltage	tolerant inputs <sup>5</sup>	-0.3	0.8	V
		all other inputs	-0.3	0.3V <sub>DD</sub>	V
I <sub>OH</sub>	High-level output current	2mA outputs <sup>6</sup>		-2	mA
		4mA outputs <sup>7</sup>		-4	mA
		6mA outputs <sup>8</sup>		-6	mA
		8mA outputs <sup>9</sup>		-8	mA
		12mA outputs <sup>10</sup>		-12	mA
I <sub>OL</sub>	Low-level output current	2mA outputs <sup>6</sup>		2	mA
		4mA outputs <sup>7</sup>		4	mA
		6mA outputs <sup>8</sup>		6	mA
		8mA outputs <sup>9</sup>		8	mA
		12mA outputs <sup>10</sup>		12	mA
T <sub>A</sub>	Operating free-air temperature	-40		+85	°C

- Notes:**
- All input parameters are valid for both inputs only and bidirectional pins.
  - All output parameters are valid for both outputs only and bidirectional pins.
  - All typical values are at V<sub>DD</sub>=3.3V, T<sub>A</sub>=25°C.
  - All voltage values are with respect to V<sub>SS</sub>.
  - Tolerant bidirectional pins: C0D7-C0D0,  $\overline{\text{CREQ0}}$ ,  $\overline{\text{CACK0}}$ ,  $\overline{\text{CSTRB0}}$ ,  $\overline{\text{CRDY0}}$ , C1D7-C1D0,  $\overline{\text{CREQ1}}$ ,  $\overline{\text{CACK1}}$ ,  $\overline{\text{CSTRB1}}$ ,  $\overline{\text{CRDY1}}$ .
  - 2 mA driving capability pins: TEST\_OUT1, TEST\_OUT2.
  - 4 mA driving capability pins: D63-D0, LD63-LD0,  $\overline{\text{CDIR0}}$ ,  $\overline{\text{CDIR1}}$ , TIMER,  $\overline{\text{INTA}}$ .
  - 6 mA driving capability pins: C0D7-C0D0,  $\overline{\text{CREQ0}}$ ,  $\overline{\text{CACK0}}$ ,  $\overline{\text{CSTRB0}}$ ,  $\overline{\text{CRDY0}}$ , C1D7-C1D0,  $\overline{\text{CREQ1}}$ ,  $\overline{\text{CACK1}}$ ,  $\overline{\text{CSTRB1}}$ ,  $\overline{\text{CRDY1}}$ .
  - 8 mA driving capability pins: A15-A1,  $\overline{\text{RAS0/CS0}}$ ,  $\overline{\text{RAS1/CS1}}$ ,  $\overline{\text{CASL/WEL}}$ ,  $\overline{\text{CASH/WEH}}$ ,  $\overline{\text{WE/A16}}$ ,  $\overline{\text{HOLD0/A17}}$ ,  $\overline{\text{HOLD1/A18}}$ ,  $\overline{\text{RDY/A19}}$ , LA15-LA1,  $\overline{\text{LRAS0/LCS0}}$ ,  $\overline{\text{LRAS1/LCS1}}$ ,  $\overline{\text{LCASL/LWEL}}$ ,  $\overline{\text{LCASH/LWEH}}$ ,  $\overline{\text{LWE/LA16}}$ ,  $\overline{\text{LHOLD0/LA17}}$ ,  $\overline{\text{LHOLD1/LA18}}$ ,  $\overline{\text{LRDY/LA19}}$ .
  - 12 mA driving capability pins:  $\overline{\text{OE}}$ ,  $\overline{\text{LOE}}$ .

### 3.4 DC Electrical Characteristics

DC electrical characteristics below are valid for the recommended operating conditions described in section 3.2 on page 39 (unless otherwise noted).

DC Electrical Characteristics

Table 3-3. DC Electrical Characteristics

Symbol	Parameter <sup>1,2</sup>	Test Conditions	Min	Nom <sup>3</sup>	Max	Unit <sup>4</sup>
V <sub>OH</sub>	High-level output voltage	V <sub>DD</sub> =Min, I <sub>OH</sub> =Max	2.4			V
		V <sub>DD</sub> =Min, I <sub>OH</sub> =-1μA	V <sub>DD</sub> -0.05			V
V <sub>OL</sub>	Low-level output voltage	V <sub>DD</sub> =Max, I <sub>OL</sub> =Max			0.4	V
		V <sub>DD</sub> =Max, I <sub>OL</sub> =1μA			0.05	V
I <sub>IH</sub>	High-level input current	input with pull-down <sup>5</sup>	V <sub>DD</sub> =Max, V <sub>IN</sub> =V <sub>DD</sub>	10	200	μA
		tolerant inputs <sup>6</sup>	V <sub>DD</sub> =Max, V <sub>IN</sub> =5V	-10	10	μA
		all other inputs	V <sub>DD</sub> =Max, V <sub>IN</sub> =V <sub>DD</sub>	-10	10	μA
I <sub>IL</sub>	Low-level input current	inputs with pull-up <sup>7</sup>	V <sub>DD</sub> =Max, V <sub>IN</sub> =V <sub>SS</sub>	-200	-10	μA
		all other inputs	V <sub>DD</sub> =Max, V <sub>IN</sub> =V <sub>SS</sub>	-10	10	μA
I <sub>OZ</sub>	Three-state output leakage current	pins with pull-up <sup>7</sup>	V <sub>DD</sub> =Max, V <sub>OUT</sub> =V <sub>DD</sub> or V <sub>SS</sub>	-200	10	μA
		all other outputs	V <sub>DD</sub> =Max, V <sub>OUT</sub> =V <sub>DD</sub> or V <sub>SS</sub>	-10	10	μA
I <sub>DD</sub>	Supply current	static	V <sub>DD</sub> =Max, T <sub>A</sub> =25°C, f <sub>CLK</sub> =0		2	mA
		idle	V <sub>DD</sub> =Max, T <sub>A</sub> =25°C, f <sub>CLK</sub> =Max <sup>8</sup>		95	mA
		dynamic	V <sub>DD</sub> =Max, T <sub>A</sub> =25°C, f <sub>CLK</sub> =Max <sup>8</sup>		300	mA
C <sub>IN</sub>	Input capacitance			7	10	pF
R <sub>PU</sub>	Pull-up resistance			100		kΩ
R <sub>PD</sub>	Pull-down resistance			100		kΩ

- Notes:**
1. All input parameters are valid for both inputs only and bidirectional pins.
  2. All output parameters are valid for both outputs only and bidirectional pins.
  3. All typical values are at V<sub>DD</sub>=3.3V, T<sub>A</sub>=25°C.
  4. All voltage values are with respect to V<sub>SS</sub>.
  5. Input with internal pull-down device: MVOE.
  6. Tolerant bidirectional pins: C0D7-C0D0,  $\overline{\text{CREQ0}}$ ,  $\overline{\text{CACK0}}$ ,  $\overline{\text{CSTRB0}}$ ,  $\overline{\text{CRDY0}}$ , C1D7-C1D0,  $\overline{\text{CREQ1}}$ ,  $\overline{\text{CACK1}}$ ,  $\overline{\text{CSTRB1}}$ ,  $\overline{\text{CRDY1}}$ .
  7. Bidirectional pins with internal pull-up devices:  $\overline{\text{RDY}}/A19$ ,  $\overline{\text{LRDY}}/LA19$ ,  $\overline{\text{CREQ0}}$ ,  $\overline{\text{CACK0}}$ ,  $\overline{\text{CSTRB0}}$ ,  $\overline{\text{CRDY0}}$ ,  $\overline{\text{CREQ1}}$ ,  $\overline{\text{CACK1}}$ ,  $\overline{\text{CSTRB1}}$ ,  $\overline{\text{CRDY1}}$ .
  8. f<sub>CLK</sub> is the input clock frequency. The maximum value is 40MHz.

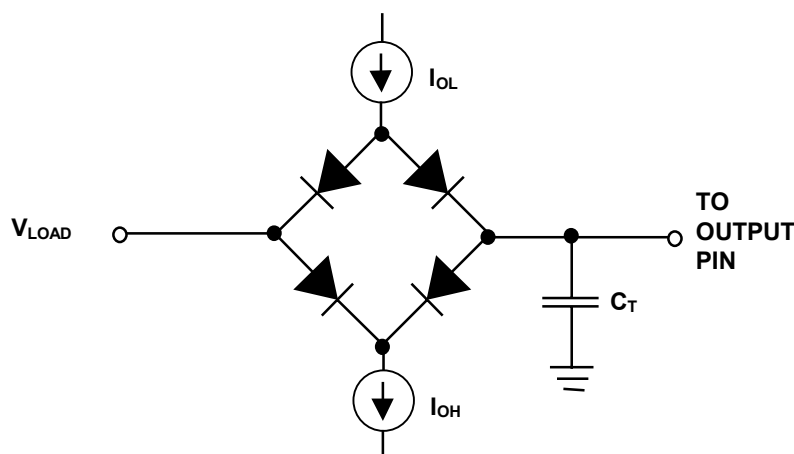
## 4 Timing Parameters

**Timing requirements** apply to signal that are controlled by circuitry external to the processor, such as the data input for a read operation. Timing requirements guarantee that the processor operates correctly with other devices.

**Switching characteristics** specify how the processor changes its signals. You have no control over this timing – circuitry external to the processor must be designed for compatibility with these signal characteristics. Switching characteristics tell you what the processor will do in a given circumstance. You can also use switching characteristics to ensure that any timing requirement of a device connected to the processor (such as memory) is satisfied.

Use the exact timing information given. Do not attempt to derive parameters from the addition or subtraction of others. While addition or subtraction would yield meaningful results for an individual device, the values given in this data sheet reflect statistical variations and worst cases. Consequently, you cannot meaningfully add up parameters to derive longer times.

The following timing requirements and switching characteristics are preliminary information and are subject to change. The test load circuit is presented in Figure 2-1.



Where:  $I_{OL} = 2.0 \text{ mA}$  (all outputs)  
 $I_{OH} = -2.0 \text{ mA}$  (all outputs)  
 $V_{LOAD} = 1.5 \text{ V}$   
 $C_T = 75 \text{ pF}$  typical load circuit capacitance

Figure 4-1. Test Load Circuit

### 4.1 Clock Signals

Table 4-2. Timing Requirements for Input Clock

Name	Description	Min	Max	Unit
$t_{c(C)}$	Cycle time, CLK	25	$\infty^1$	ns
$t_{w(CH)}$	Pulse duration, CLK high	12.5		ns
$t_{w(CL)}$	Pulse duration, CLK low	12.5		ns
$t_{f(C)}$	Fall time, CLK		5	ns
$t_{r(C)}$	Rise time, CLK		5	ns

**Note:** 1. This device is implemented in static logic and therefore can operate with  $t_{c(CK)}$  approaching  $\infty$ .

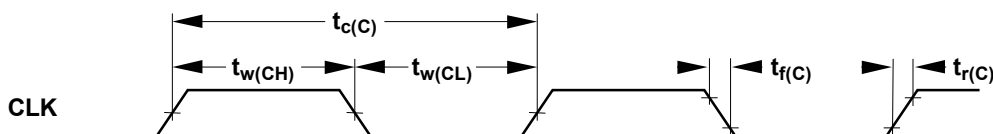


Figure 4-2. Clock Timing Diagram

## 4.2 Reset Timing

Table 4-4. Timing Requirements for Reset

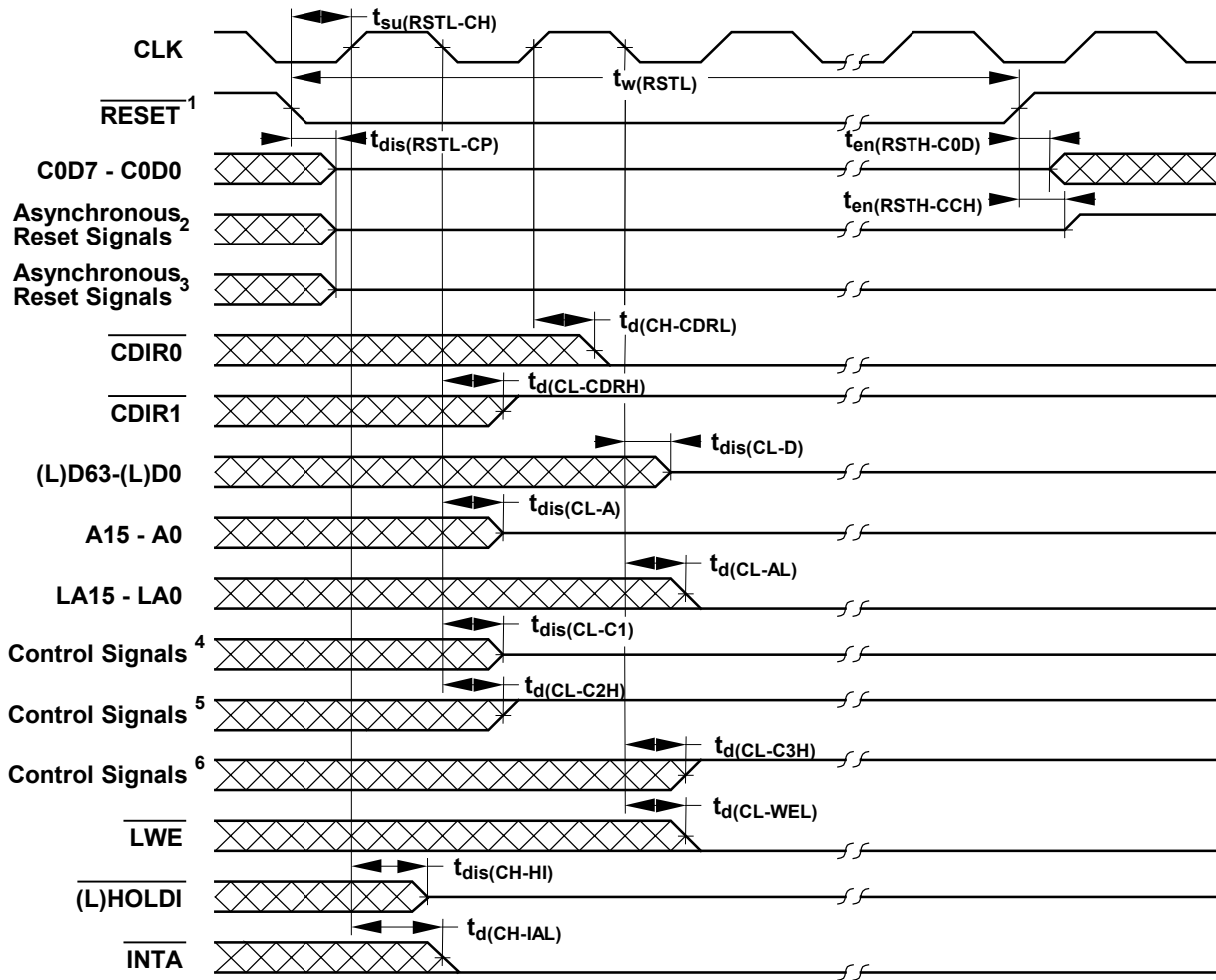
Name	Description	Min	Max	Unit
$t_{w(RSTL)}$ <sup>1</sup>	Pulse duration, $\overline{RESET}$ low	10P		ns
$t_{su(RSTL-CH)}$ <sup>2</sup>	Setup time, $\overline{RESET}$ low before CLK high	12		ns

- Notes:**
1. Applied after the power-up sequence is complete. No maximum duration limit exists.
  2. Only required if multiple NM6403s must come out of reset synchronous to CLK with program counters (PC) equal (i.e., for a SIMD system). Not required for multiple NM6403s communicating over the shared buses, because the bus arbitration logic synchronizes itself automatically after reset.
  3. P is the CLK clock period.

Table 4-5. Switching Characteristics for Reset Timing

Name	Description	Min	Max	Unit
$t_{dis(RSTL-CP)}$	Disable time, communication port pins disable after $\overline{RESET}$ low		5	ns
$t_{en(RSTH-C0D)}$	Enable time, C0D7 - C0D0 driven after $\overline{RESET}$ high		4.5	ns
$t_{en(RSTH-CCH)}$	Enable time, $\overline{CREQ1}$ , $\overline{CACK0}$ , $\overline{CSTRB0}$ , and $\overline{CRDY1}$ high after $\overline{RESET}$ high		5	ns
$t_d(CH-CDRL)$	Delay time, CLK high to $\overline{CDIR0}$ low		14	ns
$t_d(CL-CDRH)$	Delay time, CLK low to $\overline{CDIR1}$ high		16	ns
$t_{dis(CL-D)}$	Disable time, CLK low to (L)D63 - (L)D0 high-impedance		9	ns
$t_{dis(CL-A)}$	Disable time, CLK low to A15 - A0 high-impedance		12	ns
$t_d(CL-AL)$	Delay time, CLK low to LA15 - LA0 low		13	ns
$t_{dis(CL-C1)}$	Disable time, CLK low to $\overline{RAS0}$ , $\overline{RAS1}$ , $\overline{CASL}$ , $\overline{CASH}$ , $\overline{WE}$ , $\overline{OE}$ , (L)RDY, and TIMER high-impedance		12	ns
$t_d(CL-C2H)$	Delay time, CLK low to (L)RAS0, (L)RAS1, (L)CASL, and (L)CASH high		12.5	ns
$t_d(CL-C3H)$	Delay time, CLK low to $\overline{LOE}$ and (L)HOLD0 high		12	ns
$t_d(CL-WEL)$	Delay time, CLK low to $\overline{LWE}$ low		13	ns
$t_{dis(CH-HI)}$	Disable time, CLK high to (L)HOLD1 high-impedance		15.5	ns
$t_d(CH-IAL)$	Delay time, CLK high to $\overline{INTA}$ low		18.5	ns

Reset Timing



- Notes:**
1.  $\overline{RESET}$  is an asynchronous input and can be asserted at any point during a clock cycle. If the specified timings are met, the exact sequence shown occurs; otherwise, an additional delay of one clock cycle can occur.
  2. Asynchronous reset signals that go to a high logic level after  $\overline{RESET}$  returns to a high state include  $\overline{CREQ1}$ ,  $\overline{CACK0}$ ,  $\overline{CSTRB0}$ , and  $\overline{CRDY1}$ .
  3. Asynchronous reset signals that go into the high-impedance state after  $\overline{RESET}$  goes low include  $\overline{CREQ0}$ ,  $\overline{CACK1}$ ,  $\overline{CSTRB1}$ , and  $\overline{CRDY0}$ . At reset, communication port 0 becomes output, while communication port 1 become input.
  4. Control signals  $\overline{RAS0}/\overline{CS0}$ ,  $\overline{RAS1}/\overline{CS1}$ ,  $\overline{CASL}/\overline{WEL}$ ,  $\overline{CASH}/\overline{WEH}$ ,  $\overline{WE}/A16$ ,  $\overline{OE}$ ,  $(L)RDY/(L)A19$ , and  $TIMER$  go into the high-impedance state from first CLK falling edge after  $\overline{RESET}$  goes low.
  5. Control signals  $(L)RAS0/(L)CS0$ ,  $(L)RAS1/(L)CS1$ ,  $(L)CASL/(L)WEL$ , and  $(L)CASH/(L)WEH$  go to a high logic level from first CLK falling edge after  $\overline{RESET}$  goes low.
  6. Control signals  $\overline{LOE}$  and  $(L)HOLD0$  go to a high logic level from second CLK falling edge after  $\overline{RESET}$  goes low.

Figure 4-3. Reset Timing Diagram

### 4.3 External Interrupt Timing

Table 4-6. Timing Requirements for External Interrupt

Name	Description	Min	Max	Unit
$t_{su(IL-CL)}$	Setup time, $\overline{INT}$ low before CLK low	15		ns
$t_{h(CL-IL)}$	Hold time, $\overline{INT}$ low after CLK low	10		ns
$t_w(IL)$	Pulse duration, $\overline{INT}$ low	25		ns
$t_{h(CL-IL)}$	Hold time, $\overline{INT}$ low after $\overline{INTA}$ low	0		ns
$t_{su(IH-CL)}$	Setup time, $\overline{INT}$ high before CLK low	15		ns
$t_{h(CL-IH)}$	Hold time, $\overline{INT}$ high after CLK low	10		ns
$t_w(IH)$	Pulse duration, $\overline{INT}$ high	25		ns
$t_{h(CL-IH)}$	Hold time, $\overline{INT}$ high after $\overline{INTA}$ high	0		ns

Table 4-7. Switching Characteristics for External Interrupt

Name	Description	Min	Max	Unit
$t_{d(CL-IAL)}$	Delay time, CLK low to $\overline{INTA}$ low		18.5	ns
$t_{d(CL-IAH)}$	Delay time, CLK low to $\overline{INTA}$ high		16.5	ns
$t_w(IAL)$	Pulse duration, $\overline{INTA}$ low	2P-2		ns

**Note:** P is the CLK clock period.

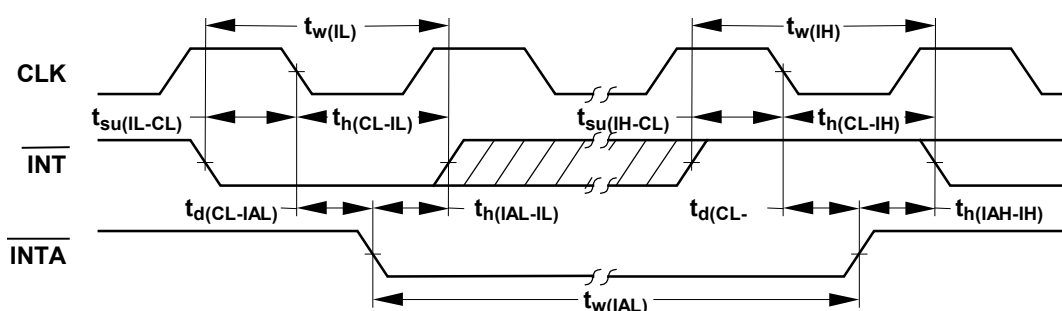


Figure 4-4. External Interrupt Timing Diagram

### 4.4 Timer Pin Timing

Period and polarity of TIMER valid logic level are specified by contents of internal control registers (refer to NM6403 Programmer's Reference Guide for more information).

Table 4-8. Switching Characteristics for Timer Pin

Name	Description	Min	Max	Unit
$t_{en(CL-T)}$	Enable time, TIMER driven after CLK low		17	ns
$t_{dis(CL-T)}$	Disable time, CLK low to TIMER high-impedance		9	ns
$t_d(CH-TL)$	Delay time, CLK high to TIMER low		13.5	ns
$t_d(CH-TH)$	Delay time, CLK high to TIMER high		15	ns

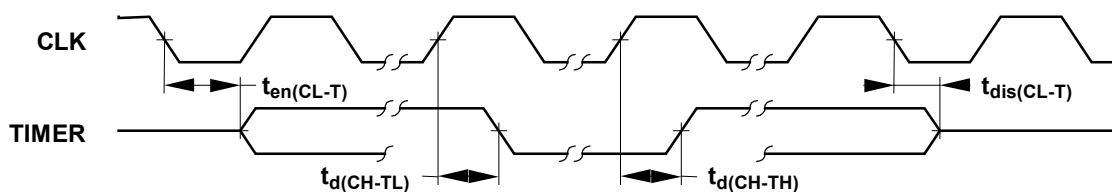


Figure 4-5. Timer Pin Timing Diagram

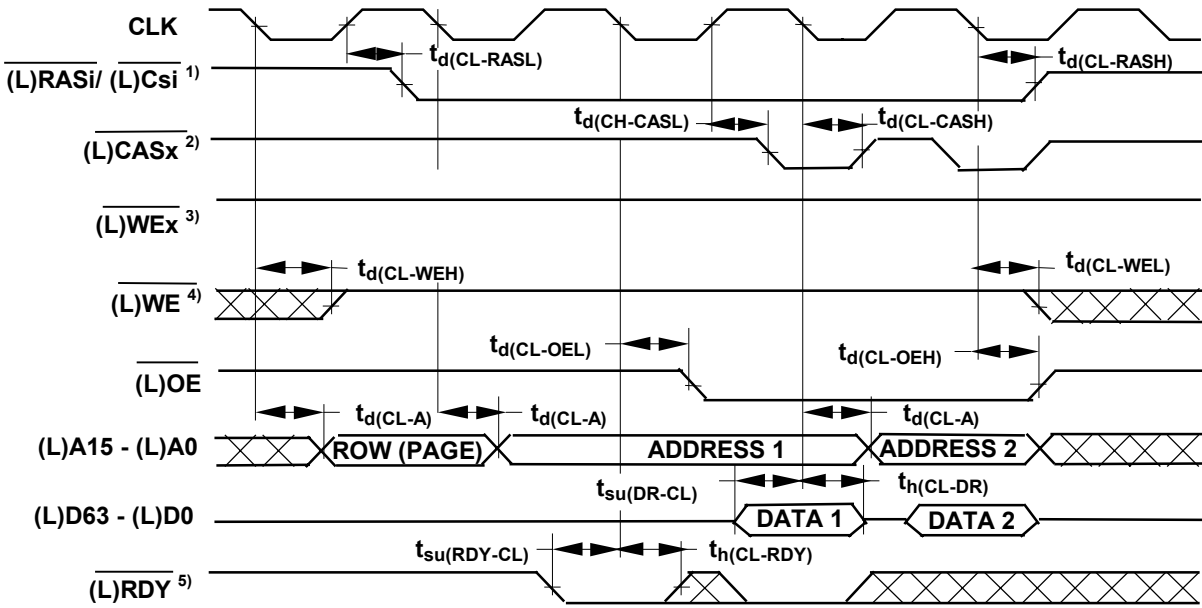
## 4.5 Memory Interface Timing

*Table 4-9. Timing Requirements for Memory Interface*

Name	Description	Min	Max	Unit
$t_{su(DR-CL)}$	Setup time, (L)D63 - (L)D0 valid before CLK low (read)	5		ns
$t_{h(CL-DR)}$	Hold time, (L)D63 - (L)D0 hold time after CLK low (read)	2		ns
$t_{su(RDY-CL)}$	Setup time, $\overline{(L)RDY}$ valid before CLK low	15		ns
$t_{h(CL-RDY)}$	Hold time, $\overline{(L)RDY}$ hold time after CLK low	10		ns

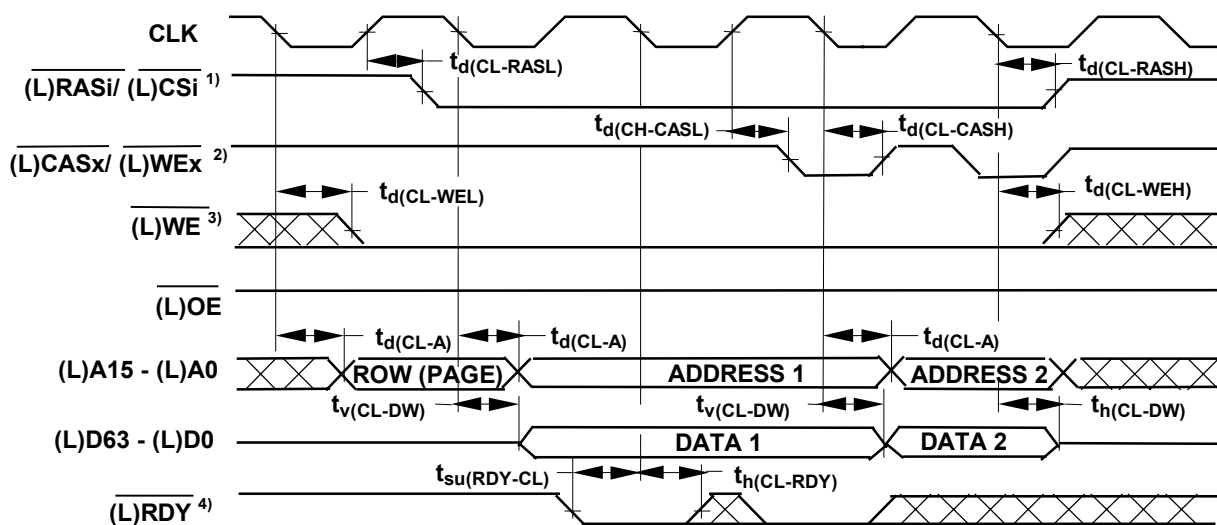
*Table 4-10. Switching Characteristics for Memory Interface*

Name	Description	Min	Max	Unit
$t_{d(CH-CASL)}$	Delay time, CLK high to $\overline{(L)CASL} / \overline{(L)WEL}$ and $\overline{(L)CASH} / \overline{(L)WEH}$ low		9.5	ns
$t_{d(CL-CASH)}$	Delay time, CLK low to $\overline{(L)CASL} / \overline{(L)WEL}$ and $\overline{(L)CASH} / \overline{(L)WEH}$ high		11.5	ns
$t_{d(CL-OEL)}$	Delay time, CLK low to $\overline{(L)OE}$ low		13.5	
$t_{d(CL-OEH)}$	Delay time, CLK low to $\overline{(L)OE}$ high		13	
$t_{d(CL-A)}$	Delay time, CLK low to (L)A15 - (L)A0 valid		12.5	
$t_{d(CL-WEL)}$	Delay time, CLK low to $\overline{(L)WE}$ low		12	
$t_{d(CL-WEH)}$	Delay time, CLK low to $\overline{(L)WE}$ high		13	
$t_{v(CL-DW)}$	(L)D63 - (L)D0 valid after CLK low (write)		18	
$t_{h(CL-DW)}$	(L)D63 - (L)D0 hold after CLK low (write)	0		
$t_{d(CH-RASL)}$	Delay time, CLK high to $\overline{(L)RAS0} / \overline{(L)CS0}$ and $\overline{(L)RAS1} / \overline{(L)CS1}$ low		10	ns
$t_{d(CL-RASH)}$	Delay time, CLK low to $\overline{(L)RAS0} / \overline{(L)CS0}$ and $\overline{(L)RAS1} / \overline{(L)CS1}$ high		11.5	ns



- Notes:**
1. It means either  $\overline{\text{(L)RAS0}}$  ( $\overline{\text{(L)RAS1}}$ ) signal for DRAM access or  $\overline{\text{(L)CS0}}$  ( $\overline{\text{(L)CS1}}$ ) signal for SRAM access.
  2. It means  $\overline{\text{CASL}}$  and  $\overline{\text{CASH}}$  signals for DRAM access, for SRAM access see Note 2.
  3. It means  $\overline{\text{(L)WEL}}$  and  $\overline{\text{(L)WEH}}$  signals for SRAM access.
  4. It means  $\overline{\text{(L)WE}}$  signal for DRAM access.
  5. In this case the  $\overline{\text{(L)RDY}}$  signal is used as external ready signal.

Figure 4-6. Memory Read Cycle Timing Diagram (Two Sequential Reads from New Page)



- Notes:**
1. It means either  $(L)RAS0$  ( $(L)RAS1$ ) signal for DRAM access or  $(L)CS0$  ( $(L)CS1$ ) signal for SRAM access.
  2. It means either  $CASL$  and  $CASH$  signals for DRAM access or  $(L)WEL$  and  $(L)WEH$  signals for SRAM access.
  3. It means  $(L)WE$  signal for DRAM access.
  4. In this case the  $(L)RDY$  signal is used as external ready input.

*Figure 4-7. Memory Write Cycle Timing Diagram (Two Sequential Writes to New Page)*

## 4.6 Communication Port Timing

All communication port timing parameters apply only to two communicating NM6403s. When a non-NM6403 device communicates with a NM6403, timings can be longer.

Communication port timing does not include line length delay.

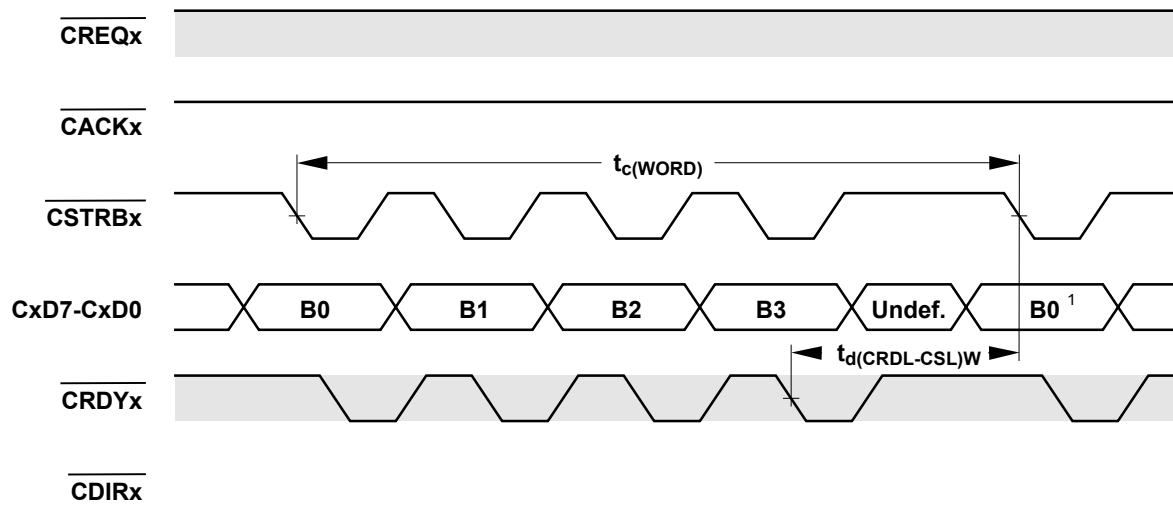
For correct operation during token exchange, the two communicating NM6403s must have CLK frequencies within a factor of 2 of each other (in other words, at most, one of the NM6403s can be twice as fast as the other).

### 4.6.1 Communication Port Word-Transfer Timing

Table 4-9. Switching Characteristics for Communication Port Word-Transfer<sup>1</sup>

Name	Description	Min	Max	Unit
$t_{c(WORD)}$ <sup>2</sup>	Cycle time, word transfer (4 bytes = 1 word)	2P	2.5P+284	ns
$t_{d(CRDL-CSL)W}$	Delay time, $\overline{CRDYx}$ low to $\overline{CSTRBx}$ low between back-to-back write cycles	1.5P	2.5P+20	ns

- Notes:**
1. P is the CLK clock period.
  2. For these timing values, it is assumed that the receiving NM6403 is ready to receive data.
  3.  $t_{c(WORD)}$  max = 2.5P + the maximum summed values of  $4 \times t_{d(CSL-CRDL)R}$ ,  $3 \times t_{d(CRDL-CSH)W}$ ,  $3 \times t_{d(CSH-CRDH)R}$ , and  $3 \times t_{d(CRDH-CSL)W}$ , as seen in Figure 4-. This assumes two NM6403s are connected.



■ = When signal is an input (clear = when signal is an output)

**Note:** 1. Begins byte 0 of the next word.

Figure 4-8. Communication Port Word-Transfer Cycle Timing Diagram

### 4.6.2 Communication Port Byte-Transfer Timing (Write and Read)

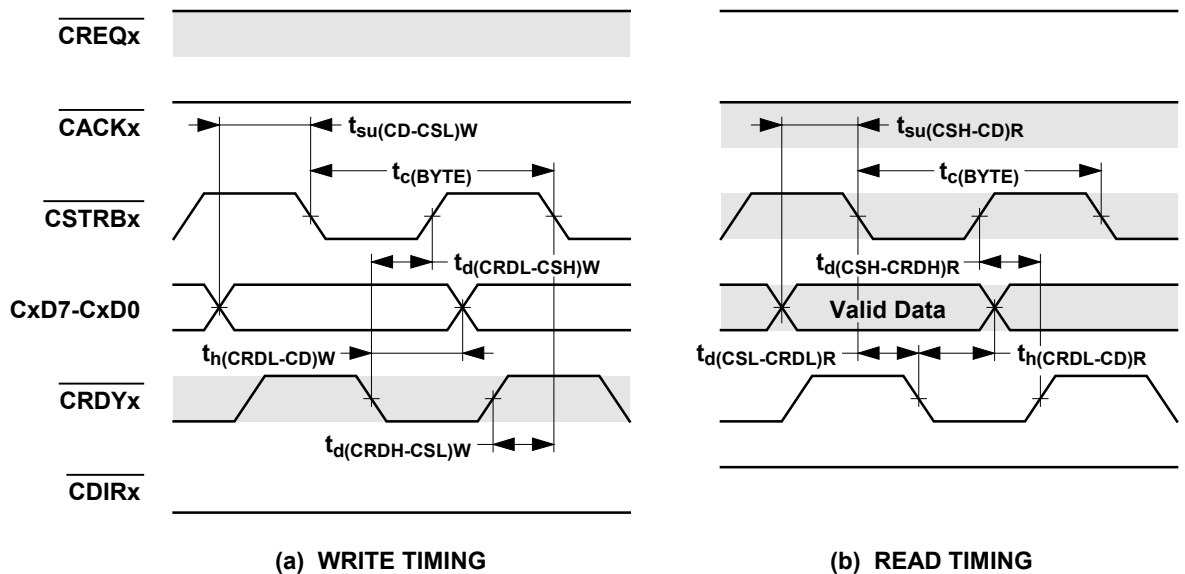
Table 4-10. Timing Requirements for Communication Port Byte-Transfer

Name	Description	Min	Max	Unit
$t_{su(CSH-CD)R}$	Setup time, CxDx valid before $\overline{CSTRBx}$ low (read)	0		ns
$t_{h(CRDL-CD)R}$	Hold time, CxDx valid after $\overline{CRDYx}$ low (read)	2		ns

Table 4-11. Switching Characteristics for Communication Port Byte-Transfer

Name	Description	Min	Max	Unit
$t_{su(CD-CSL)W}$	Setup time, CxDx valid before $\overline{CSTRBx}$ low (write)	2		ns
$t_{d(CRDL-CSH)W}$	Delay time, $\overline{CRDYx}$ low to $\overline{CSTRBx}$ high (write)	0	20	ns
$t_{h(CRDL-CD)W}$	Hold time, CxDx valid after $\overline{CRDYx}$ low (write)	2		ns
$t_{d(CRDH-CSL)W}$	Delay time, $\overline{CRDYx}$ high to $\overline{CSTRBx}$ low for subsequent bytes (write)	0	18	ns
$t_{c(BYTE)}^1$	Cycle time, byte transfer		71	ns
$t_{d(CSL-CRDL)R}$	Delay time, $\overline{CSTRBx}$ low to $\overline{CRDYx}$ low (read)	0	18	ns
$t_{d(CSH-CRDH)R}$	Delay time, $\overline{CSTRBx}$ high to $\overline{CRDYx}$ high (read)	0	15	ns

**Note:** 1.  $t_{c(BYTE)}$  max = summed maximum values of  $t_{d(CRDL-CSH)W}$ ,  $t_{d(CSH-CRDH)R}$ ,  $t_{d(CRDH-CSL)W}$ , and  $t_{d(CSL-CRDL)R}$ . This assumes two NM6403s are connected.



■ = When signal is an input (clear = when signal is an output)

Figure 4-9. Communication Port Byte-Transfer Timing Diagram (Write and Read)

4.6.3 Communication Token Transfer Sequence, Input to an Output Port

Before the token exchange,  $\overline{\text{CREQx}}$  and  $\overline{\text{CRDYx}}$  are asserted by the NM6403 receiving data.  $\overline{\text{CACKx}}$ ,  $\overline{\text{CSTRBx}}$ , and CxD7-CxD0 are input signals asserted by the NM6403 sending data and are asynchronous with respect to the CLK signal of the receiving NM6403. After token exchange,  $\overline{\text{CACKx}}$ ,  $\overline{\text{CSTRBx}}$ , and CxD7-CxD0 become outputs, and  $\overline{\text{CREQx}}$  and  $\overline{\text{CRDYx}}$  become inputs.

Table 4-12. Switching Characteristics for Communication Token Transfer Sequence, Input to an Output Port

Name	Description	Min	Max	Unit
$t_{en(\text{CAL-CS})T}^1$	Enable time, $\overline{\text{CSTRBx}}$ change from an input to a high-level output after $\overline{\text{CACKx}}$ low	0.5P	1.5P+10	ns
$t_{d(\text{CAL-CRQH})T}^1$	Delay time, $\overline{\text{CACKx}}$ low to $\overline{\text{CREQx}}$ high	P	2P+13.5	ns
$t_{dis(\text{CRQH-CRQ})T}$	Disable time, $\overline{\text{CREQx}}$ change from a high-level output to an input after $\overline{\text{CREQx}}$ high	0.5P	0.5P+9	ns
$t_{en(\text{CRQH-CA})T}$	Enable time, $\overline{\text{CACKx}}$ change from an input to a high-level output after $\overline{\text{CREQx}}$ high	0.5P	0.5P+9	ns
$t_{en(\text{CRQH-CD})T}$	Enable time, CxDx change from inputs to outputs after $\overline{\text{CREQx}}$ high	0.5P	0.5P+8	ns
$t_{dis(\text{CRQH-CRD})T}$	Disable time, $\overline{\text{CRDYx}}$ change from a high-level output to an input after $\overline{\text{CREQx}}$ high	0.5P	0.5P+9	ns
$t_{d(\text{CRQH-CSL})T}$	Delay time, $\overline{\text{CREQx}}$ high to $\overline{\text{CSTRBx}}$ low for start of word transfer out	1.5P	1.5P+20	ns
$t_{d(\text{CRDL-CSL})T}^1$	Delay time, $\overline{\text{CRDYx}}$ low at end of word input to $\overline{\text{CSTRBx}}$ low for word output	3.5P	5.5P+20	ns
$t_{d(\text{CRQH-CDRL})T}$	Delay time, $\overline{\text{CREQx}}$ high to $\overline{\text{CDIRx}}$ low	0.5P	0.5P+14.5	ns

Note: 1. These timing parameters result from synchronizer delays (P is the CLK clock period).

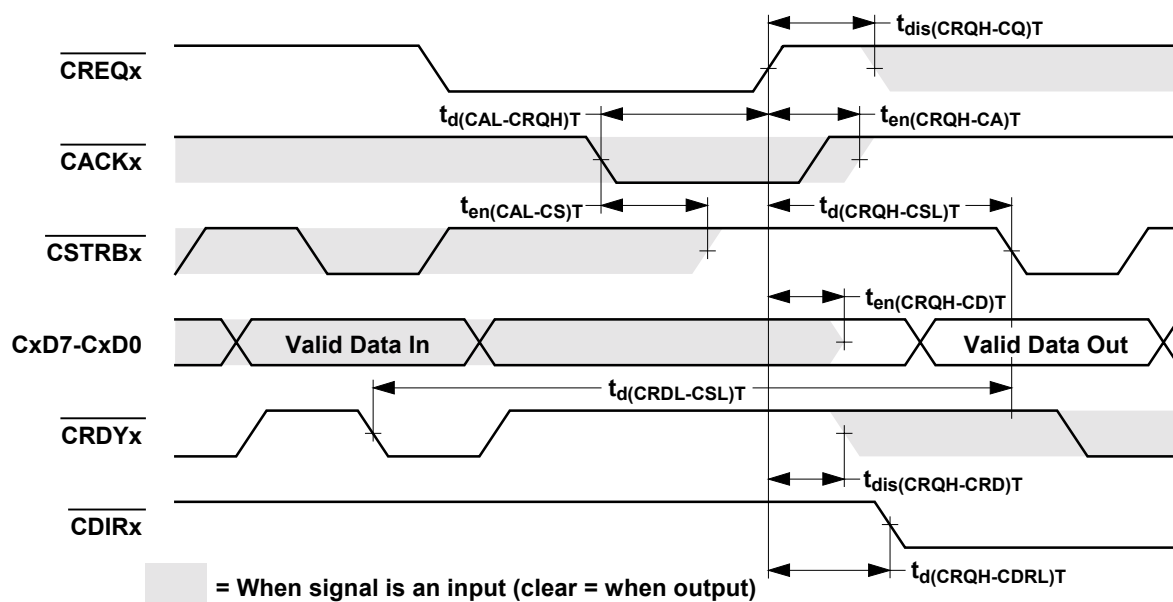


Figure 4-10. Communication Token Transfer Sequence, Input to an Output Port

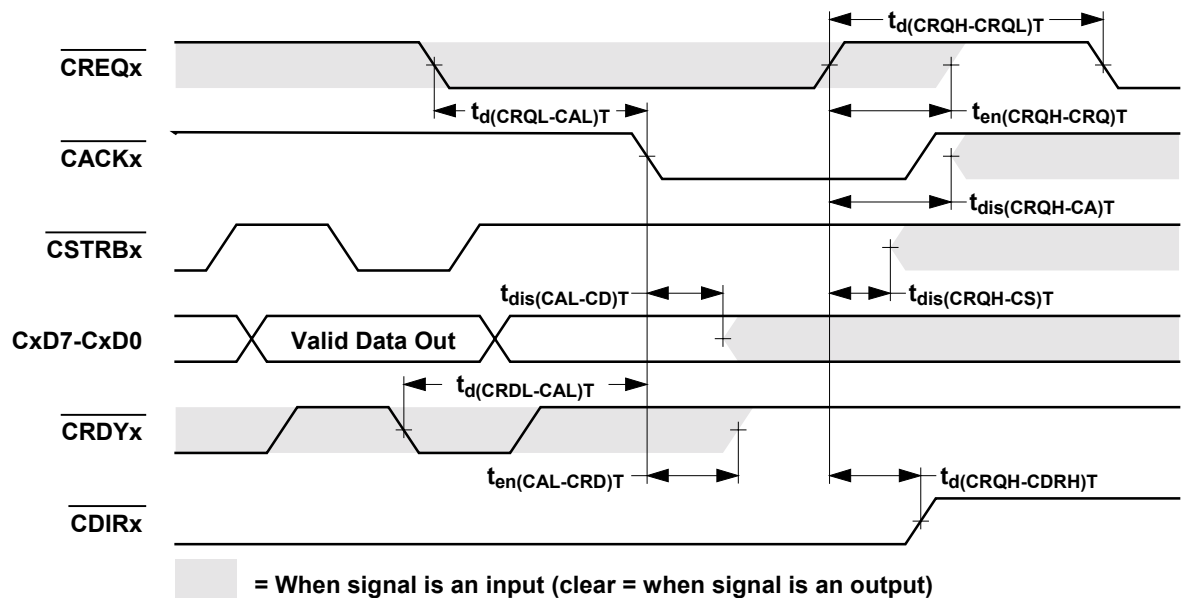
#### 4.6.4 Communication Token Transfer Sequence, Output to an Input Port

Before the token exchange,  $\overline{\text{CACKx}}$ ,  $\overline{\text{CSTRBx}}$ , and CxD7-CxD0 are asserted by the NM6403 sending data.  $\overline{\text{CREQx}}$  and  $\overline{\text{CRDYx}}$  are input signals asserted by the NM6403 receiving data and are asynchronous with respect to the CLK signal of the sending NM6403. After token exchange,  $\overline{\text{CREQx}}$  and  $\overline{\text{CRDYx}}$  become outputs, and  $\overline{\text{CACKx}}$ ,  $\overline{\text{CSTRBx}}$ , and CxD7-CxD0 become inputs.

*Table 4-13. Switching Characteristics for Communication Token Transfer Sequence, Output to an Input Port*

Name	Description	Min	Max	Unit
$t_{d(\text{CRQL-CAL})T}^1$	Delay time, $\overline{\text{CREQx}}$ low to $\overline{\text{CACKx}}$ low	P	2P+16	ns
$t_{d(\text{CRDL-CAL})T}^1$	Delay time, $\overline{\text{CRDYx}}$ low at end of world transfer out to $\overline{\text{CACKx}}$ low	P	2P+16	ns
$t_{\text{dis}(\text{CAL-CD})I}$	Disable time, CxDx change from outputs to inputs after $\overline{\text{CACKx}}$ low	0.5P	0.5P+7.5	ns
$t_{\text{en}(\text{CAL-CRD})T}$	Enable time, $\overline{\text{CRDYx}}$ change from an input to a high-level output after $\overline{\text{CACKx}}$ low	0.5P	0.5P+8	ns
$t_{\text{en}(\text{CRQH-CRQ})T}$	Enable time, $\overline{\text{CREQx}}$ change from an input to a high-level output after $\overline{\text{CREQx}}$ high		18	ns
$t_{\text{dis}(\text{CRQH-CA})T}$	Disable time, $\overline{\text{CACKx}}$ change from a high-level output to an input after $\overline{\text{CREQx}}$ high		18	ns
$t_{\text{dis}(\text{CRQH-CS})T}$	Disable time, $\overline{\text{CSTRBx}}$ change from a high-level output to an input after $\overline{\text{CREQx}}$ high		9	ns
$t_{d(\text{CRQH-CRQL})T}^1$	Delay time, $\overline{\text{CREQx}}$ high to $\overline{\text{CREQx}}$ low for next token request	P	2P+15.5	ns
$t_{d(\text{CRQH-CDRH})T}$	Delay time, $\overline{\text{CREQx}}$ high to $\overline{\text{CDIRx}}$ high		13.5	ns

**Note:** 1. These timing parameters result from synchronizer delays (P is the CLK clock period).



*Figure 4-11. Communication Token Transfer Sequence, Output to an Input Port*

## 5 Power Dissipation

This section describes the equations on how to estimate the power dissipation for NM6403. The following characteristics are preliminary and are subject to change.

The total power dissipation ( $P_{TOTAL}$ ) consists of static power dissipation ( $P_{DC}$ ) and dynamic power dissipation ( $P_{AC}$ ):  $P_{TOTAL} = P_{DC} + P_{AC}$ .

$P_{DC} = P_{DC\_OUT}$ , where  $P_{DC}$  output is the static power when output buffers source or sink.

$P_{AC} = P_{AC\_INPUT} + P_{AC\_OUTPUT} + P_{AC\_INTERNAL}$ , where  $P_{AC\_INPUT}$  and  $P_{AC\_OUTPUT}$  - input and output buffers dynamic power dissipation,  $P_{AC\_INTERNAL}$  - internal cells dynamic power dissipation.

$$P_{DC\_OUT} = 150 \cdot I_{OL} \cdot N_{OUTPUT} \text{ [}\mu\text{W]};$$

$$P_{AC\_INPUT} = 10.8 \cdot N_{INPUT} \cdot F \cdot S \text{ [}\mu\text{W]};$$

$$P_{AC\_OUTPUT} = 11 \cdot N_{OUTPUT} \cdot F \cdot S \cdot C \text{ [}\mu\text{W]};$$

$$P_{AC\_INTERNAL} = 149,500 \cdot F \cdot S \text{ [}\mu\text{W]};$$

where  $I_{OL}$  is source and sink current of output buffers in mA (see Table 2-3),

$N_{OUTPUT}$  is the number of output buffers used,

$N_{INPUT}$  is the number of input buffers used,

$F$  is the input clock frequency in MHz (the maximum value is 40 MHz),

$S$  is the estimated degree of a switching activity (typically 0.2),

$C$  is the output load capacitance in pF.

The total power dissipation,  $P_{TOTAL}$  may be used to determine the maximum junction temperature ( $T_J$ ) for the device as follows:

$$(T_J = P_{TOTAL} \cdot \Theta_{JA} + 85^\circ\text{C}) \leq 125^\circ\text{C},$$

where  $\Theta_{JA}$  is the thermal impedance of package = 24 °C/W.

## 6 Mechanical Data

The NM6403 is offered in 27-mm 256-pin PBGA package. The pitch of the solder balls is 1.27 mm. Refer to the following figure for package drawing and dimensions. All linear dimensions are in millimeters.

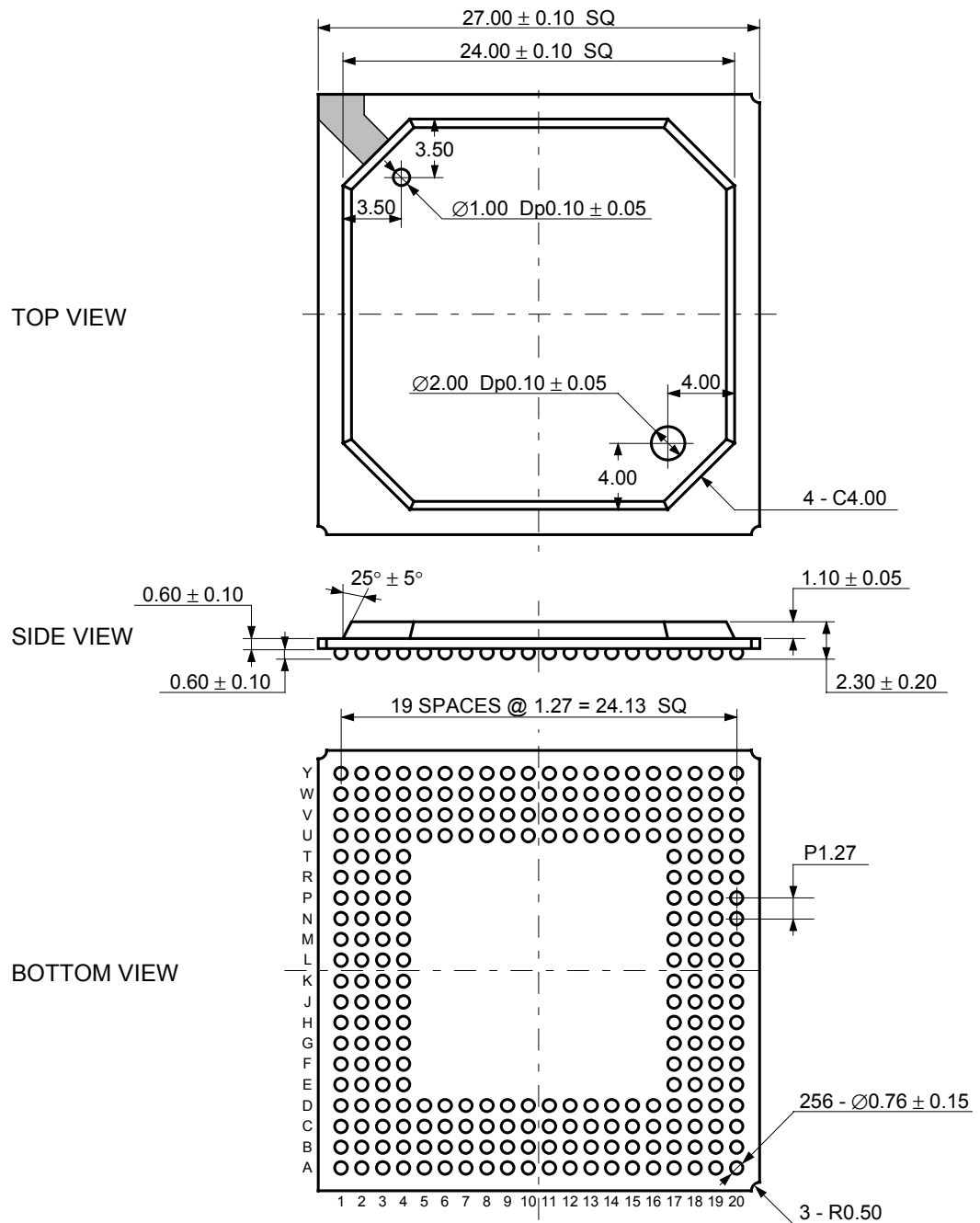


Figure 6-1. PBGA Package Mechanical Drawing

## 7 Design Considerations

### 7.1 Electrical Design Considerations

Use the following list of recommendations to assure correct NM6403 operation:

- Provide a low-impedance path from the board power supply to each VDD pin on the NM6403, and from the board ground to each VSS pin.
- Use at least six 0.1  $\mu$ F bypass capacitors positioned as close as possible to the package supply pins, one capacitor for two VDD/VSS pairs. The recommended bypass configuration is to place one bypass capacitor on each of the twelve VDD/VSS pairs.
- Ensure that capacitor leads and associated printed circuit traces that connect to the chip VDD and VSS pins are less than 12 mm per capacitor lead.
- Use at least a four-layer Printed Circuit Board (PCB) with two inner layers for V<sub>DD</sub> and V<sub>SS</sub>.
- Bypass the V<sub>DD</sub> and V<sub>SS</sub> layers of the PCB with approximately 100  $\mu$ F, preferably with a high-grade capacitor such as a tantalum capacitor.
- Because the NM6403 output signals have fast rise and fall times, PCB trace lengths should be minimal.
- Consider all device loads as well as parasitic capacitance due to PCB traces when calculating capacitance. This is especially critical in systems with higher capacitive loads that could create higher transient currents in the V<sub>DD</sub> and V<sub>SS</sub> circuits.
- All input must be terminated (i.e., not allowed to float) using CMOS levels. Unused inputs should be tied to an appropriate logic voltage level (e.g., V<sub>DD</sub> or V<sub>SS</sub>). Unused bidirectional pins should be tied to external pull-up or pull-down resistors, except for pins that have internal pull-up or pull-down resistors ( $\overline{\text{RDY}}/\text{A19}$ ,  $\overline{\text{LRDY}}/\text{LA19}$ ,  $\overline{\text{CREQ0}}$ ,  $\overline{\text{CACK0}}$ ,  $\overline{\text{CSTRB0}}$ ,  $\overline{\text{CRDY0}}$ ,  $\overline{\text{CREQ1}}$ ,  $\overline{\text{CACK1}}$ ,  $\overline{\text{CSTRB1}}$ ,  $\overline{\text{CRDY1}}$ ) - these pins should be left floating. These pins have a logic level hold circuit that prevents the input from floating internally.
- BOOTM input must be tied to V<sub>SS</sub> or V<sub>DD</sub> depending on desired NM6403 boot mode. If BOOTM is set to 0, the processor is initialized from global bus, if set to 1 the processor is initialized from communication port 1.
- TIMER should be tied to external pull-up or pull-down resistor. If the processor is initialized from DRAM on the global bus, external pull-up should be used. If the processor is initialized from SRAM on the global bus, external pull-down should be used. For initialization from communication port 1 it is don't care.
- TEST\_OUT1, TEST\_OUT2 and MVOE pins are designed for manufacturer testing purposes only. These pins should be left floating in application designs.

## 7.2 Development support

The NM6404 is supported with a complete set of software and hardware development tools, including tools to evaluate the performance of the DSP, generate code, develop algorithm implementations, and fully integrate and debug software and hardware modules.

The following products support development of NM6403-based applications:

### Software Development Tools:

- C++ compiler,
- Assembler,
- Linker,
- Instruction level and cycle accurate simulator,
- Source level debugger,
- Librarian,
- Load and exchange library,
- Set of system and applied libraries.

### Hardware Development Tools:

- NeuroMatrix® NM1 Dual-DSP PCI Board,
- NeuroMatrix® NM2 DSP PCI Board,
- NeuroMatrix® NM4 Quad-DSP CompactPCI Board,
- NeuroMatrix® NMT403 TIM.

Additional details are available in the *NM6403 Programmer's Reference Guide*.

## 7.3 Product Documentation

The four manuals listed in Table 7-1 are required for complete description of the NM6403 and are necessary to design properly with the part. Documentation is available from a RC *MODULE* office or the RC *MODULE* home page on the World Wide Web.

For the latest information refer to the back cover of this document for RC *MODULE* contact address and RC *MODULE* internet address.

*Table 7-1. NM6403 Documentation*

Document Name	Description of Contents
NM6403 Data Sheet	General overview of architecture and functionality, electrical and timing specifications, pin and package descriptions
NM6403 User's Manual	Detailed description of architecture, instruction set, peripherals, and interfaces
NM6403 Programmer's Reference Guide	Detailed description of programming model, assembly language, and software development tools
NM6403 Programmer's Tutorial	Description of step by step software development



**JOINT-STOCK COMPANY  
RESEARCH CENTRE**

**Research Centre Module**

**Box: 166, Moscow, 125190, Russia**

**Tel: +7 (095) 152-9335**

**Fax: +7 (095) 152-4661**

**E-Mail: [info@module.ru](mailto:info@module.ru)**

**WWW: <http://www.module.ru>**

©RC Module, 2000

All rights reserved.

Neither the whole nor any part of the information contained in, or the product described in this overview may be adapted or reproduced in any form except with the prior written permission of the copyright holder.

RC Module reserves the right to make changes without further notice to product herein to improve reliability, function or design. RC Module shall not be liable for any loss or damage arising from the use of any information in this overview or any error or omission in such information, or any incorrect use of the product.

Printed in Russia

Data of release: May 2000