Національний університет "Львівська політехніка"
Кафедра електронних обчислювальних машин (ЕОМ)

Автоматизоване проектування комп'ютерних систем

спеціальність 123 “Комп’ютерна інженерія”
спеціалізація 123.01 “Комп’ютерні системи”
4-ий курс

Лекція 4. Високорівневі засоби системного проектування (2)

2021

Лекція 4. Високорівневі засоби системного проектування (2)

01. Системна шина WISHBONE.
02. Типи з’єднань на шині WISHBONE.
03. Інтерфейсні сигнали WISHBONE.
04. Цикли шини WISHBONE.

01.1. Системна шина Wishbone.

The WISHBONE System-on-Chip (SoC) interconnection є методом з’єднання цифрових схем разом в формі інтегрованого чіпу (form an integrated circuit ‘chip’). Архітектура WISHBONE вирішує фундаментальні проблеми проектування інтегральних схем, а саме, як з’єднати поміж собою схемно реалізовані функції системи в спосіб, що є нескладним, гнучким і портативним (пересувним). Реалізовані функції залучають до проекту в формі так званих IP-ядер (‘IP Cores’ - Intellectual Property Cores), які системні інтегратори(розробники систем на кристалі) або купують, або самостійно розробляють.

01.2. Системна шина Wishbone.

З використанням архітектури WISHBONE IP-ядра фактично перетворюються на “будівельні блоки”, з яких монтують систему. Цими блоками є мікропроцесори, послідовні порти, дискові інтерфейси, мережеві контролери тощо. Це можливо лише за умови стандартизації інтерфейсу до IP-ядер. Саме цій стандартизації і сприяє шина WISHBONE.

02.1. Типи з’єднань на шині WISHBONE.

Типи з’єднань на шині WISHBONE:

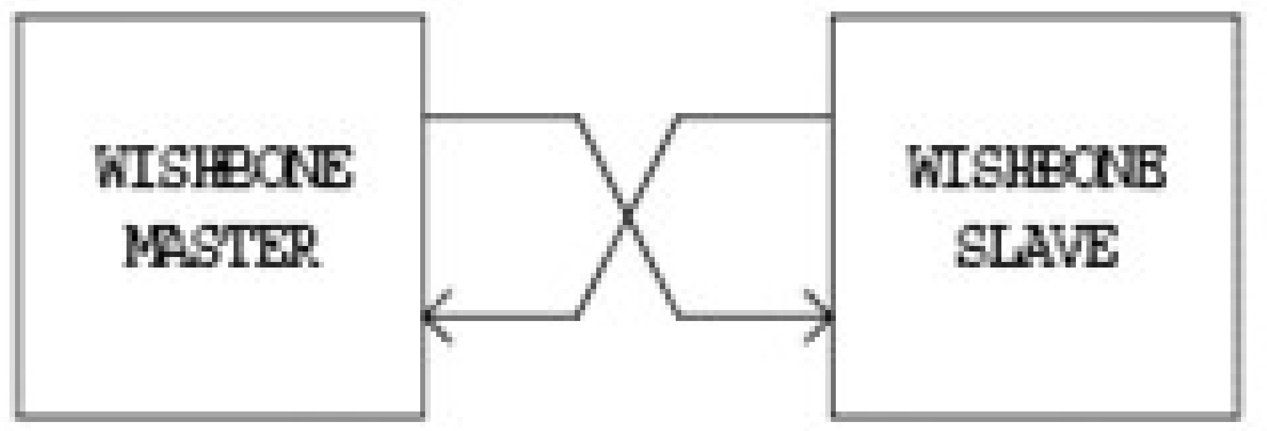
- з’єднання точка-точка (point-to-point);
- з’єднання потоком даних (Data Flow);
- з’єднання розділеною шиною (Shared Bus);
- з’єднання перехрестним комутатором (Crossbar Switch).

02.2. Типи з’єднань на шині WISHBONE.

З’єднання точка-точка (point-to-point)
З’єднання “точка-точка”- це найпростіший спосіб з'єднати між собою два ядра. Взаємозв'язок "точка-точка" дозволяє одному інтерфейсу MASTER підключатися до одного інтерфейсу SLAVE. Наприклад, інтерфейс MASTER може бути на IP-ядрі мікропроцесора , а інтерфейс SLAVE - на послідовному порту вводу-виводу.

02.3. Типи з’єднань на шині WISHBONE.

З’єднання точка-точка (point-to-point)



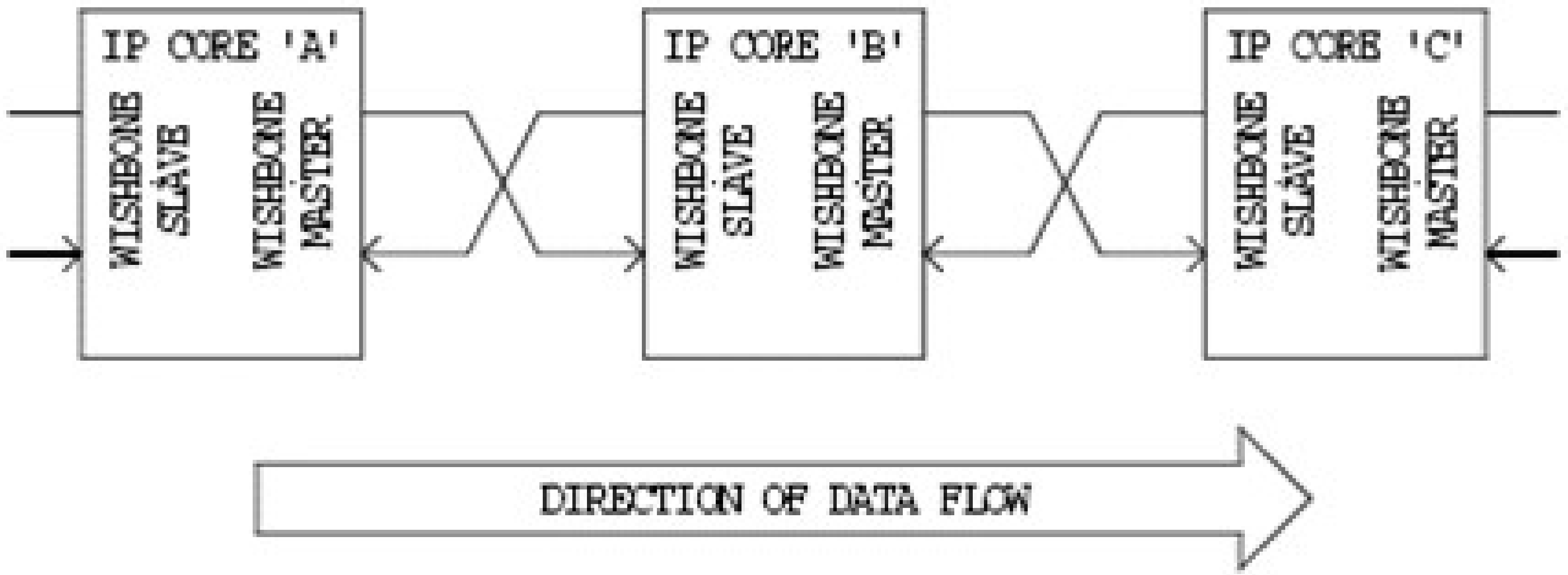
02.4. Типи з’єднань на шині WISHBONE.

З’єднання потоку даних (Data Flow)

З’єднання потоку даних використовується, коли дані обробляються послідовно. Кожне IP-ядро в архітектурі потоку даних має як MASTER, так і SLAVE-інтерфейс. Передача даних відбувається від ядра до ядра. Таку систему також можна вважати конвеєром.

02.5. Типи з’єднань на шині WISHBONE.

З’єднання потоку даних (Data Flow)

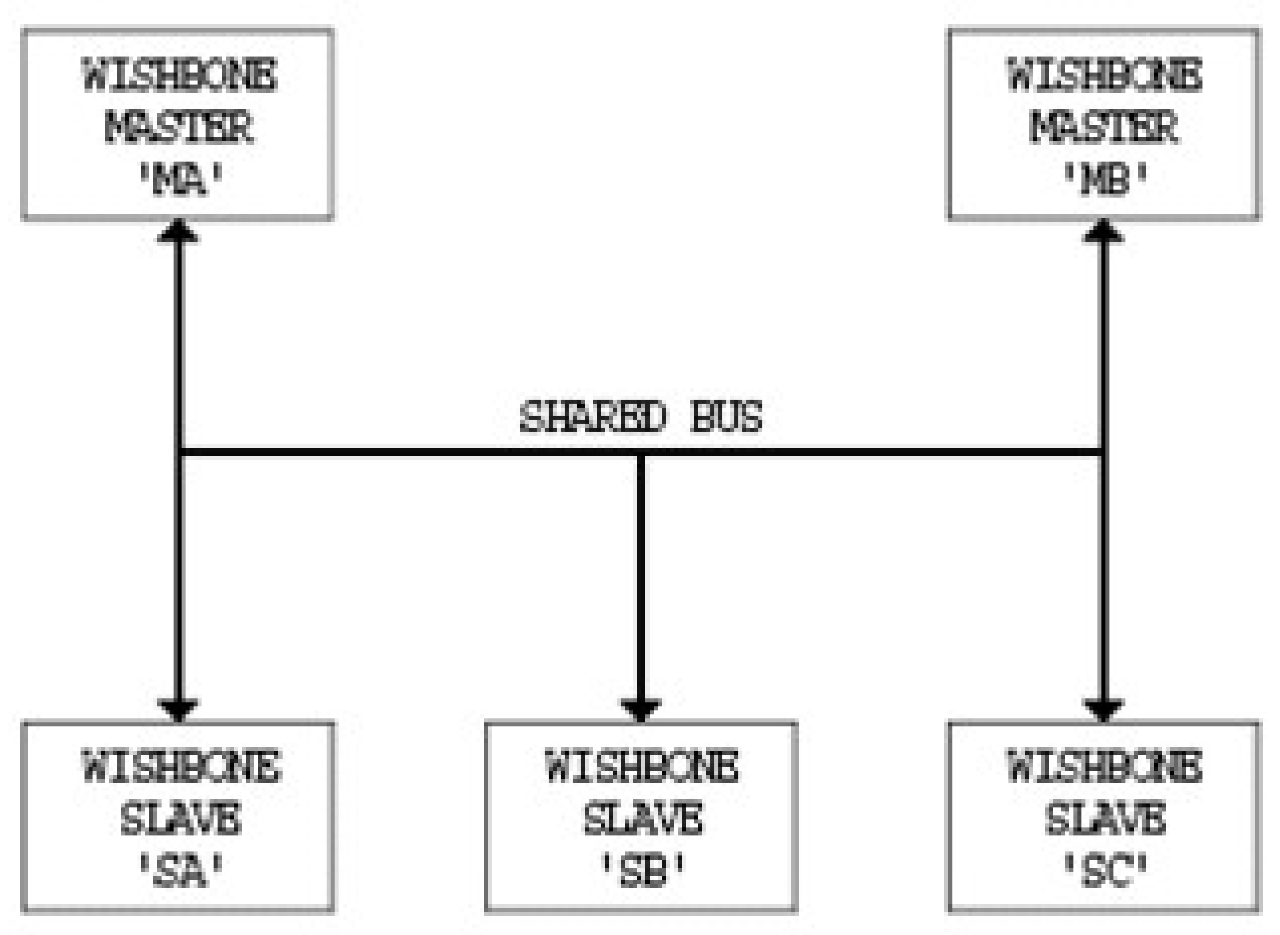


02.6. Типи з’єднань на шині WISHBONE.

З’єднання спільною шиною (Shared Bus)
Спільне з'єднання корисно для з'єднання двох або більше MASTER з одним або декількома SLAVE. У цій топології MASTER ініціює цикл шини до цільового SLAVE. Цільовий SLAVE тоді виконує один або кілька циклів шини разом з MASTER.
Арбітр (не показано далі на рисунку) визначає, коли MASTER може отримати доступ до спільної шини. Арбітр визначає коли і як кожен MASTER звертається до спільного ресурсу. Крім того, тип арбітра повністю визначається системним інтегратором. Наприклад, спільна шина може використовувати пріоритетний або круговий арбітр. Вони надають спільну шину на пріоритетній або рівній основі відповідно.

02.7. Типи з’єднань на шині WISHBONE.

З’єднання спільною шиною (Shared Bus)

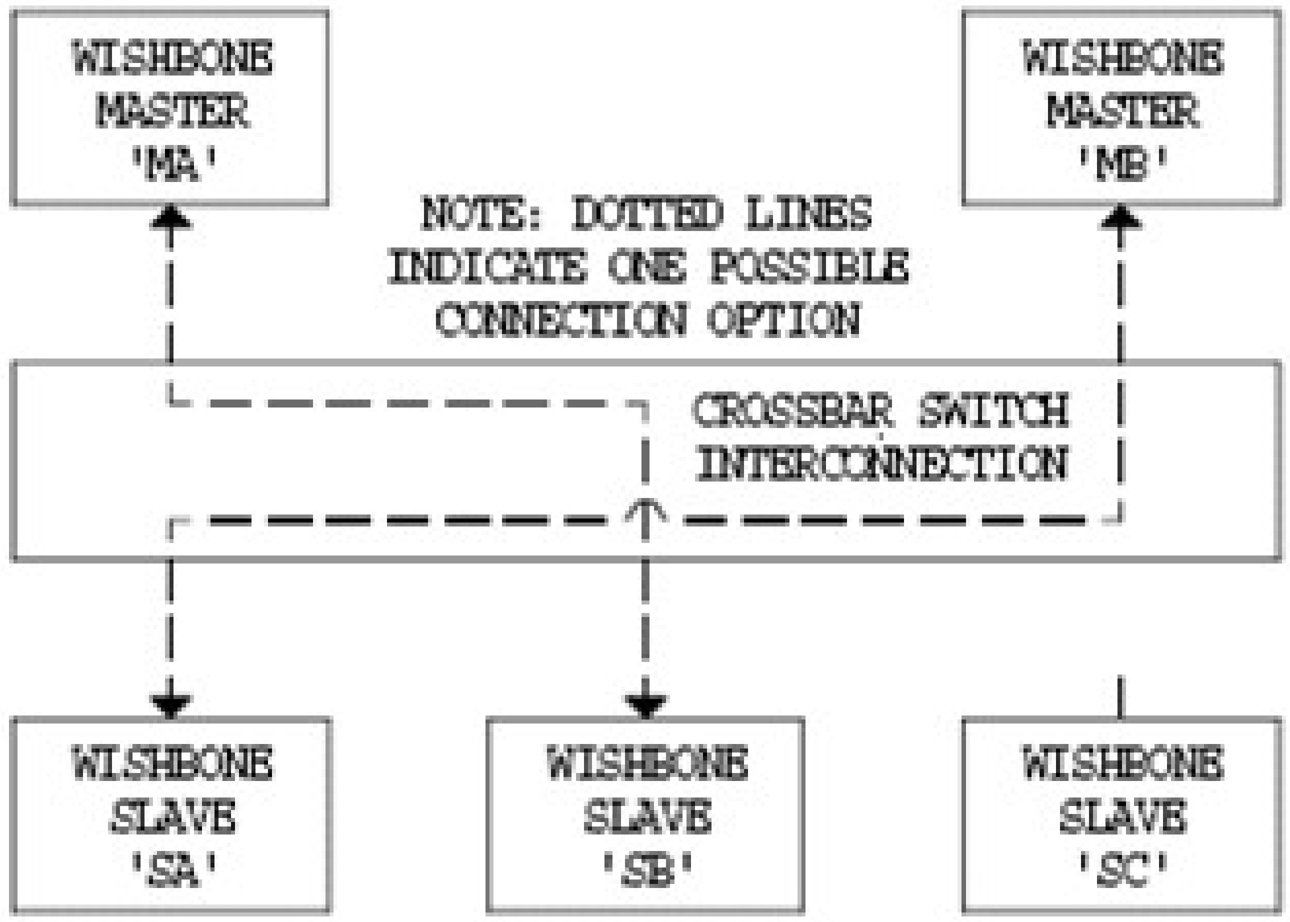


02.8. Типи з’єднань на шині WISHBONE.

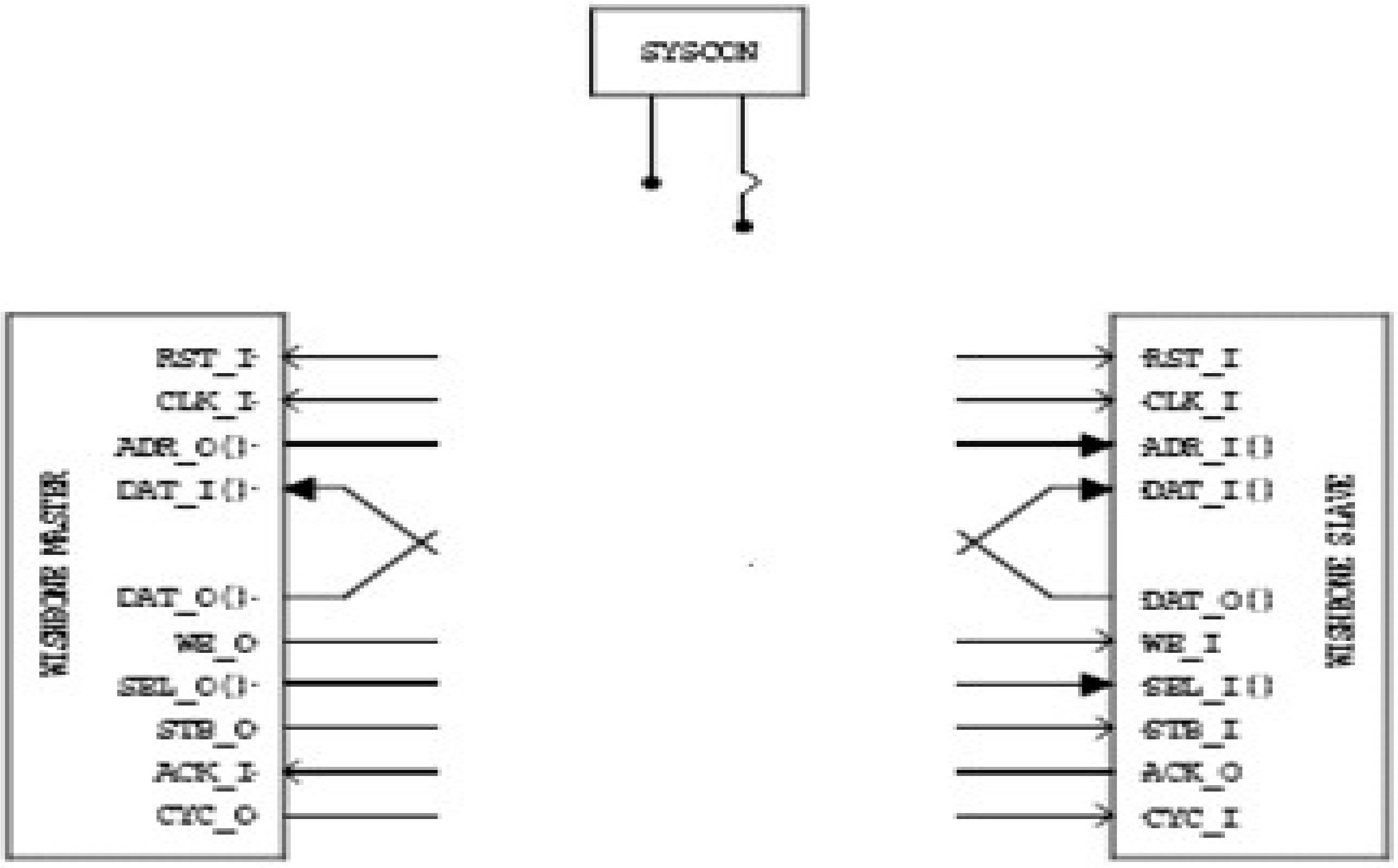
З’єднання перехресним комутатором (Crossbar Switch) Взаємозв'язок на основі перехресного комутатора використовується при з'єднанні двох або більше MASTER разом, щоб кожен міг отримати доступ до двох або більше SLAVE. MASTER ініціює цикл шини для цільового SLAVE. Арбітр (не показано далі на рисунку) визначає, коли кожен MASTER може отримати доступ до зазначеного SLAVE. На відміну від спільного з'єднання шини, тут можна використовувати більше ніж один MASTER (до тих пір, поки два MASTER не мають доступу той самого SLAVE одночасно).

02.9. Типи з’єднань на шині WISHBONE.

З’єднання перехрестним комутатором (Crossbar Switch)

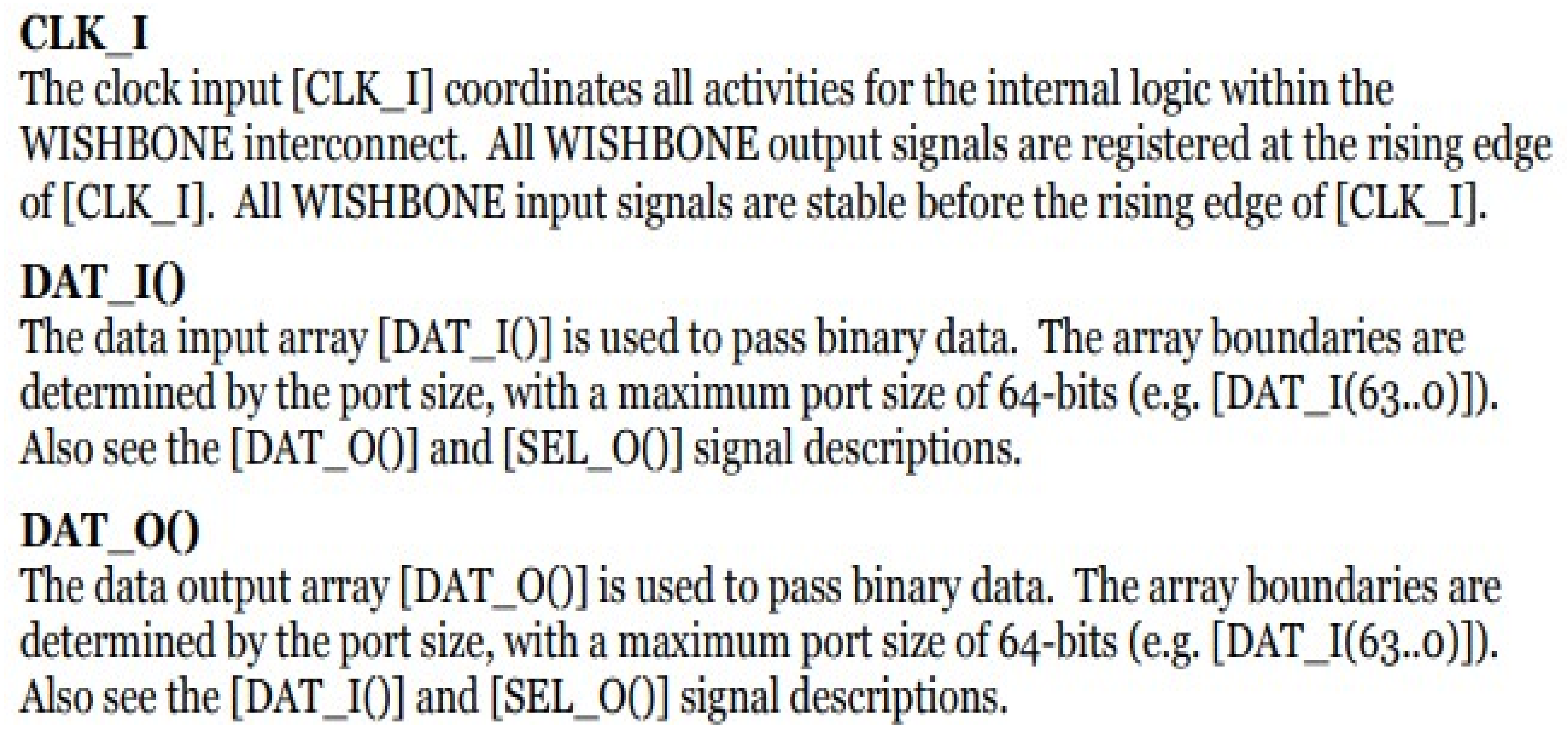


03.1. Інтерфейсні сигнали Wishbone.



FORMING A POINT-TO-POINT INTERCONNECTION.

03.2. Інтерфейсні сигнали Wishbone.
Signals Common to MASTER and SLAVE Interfaces

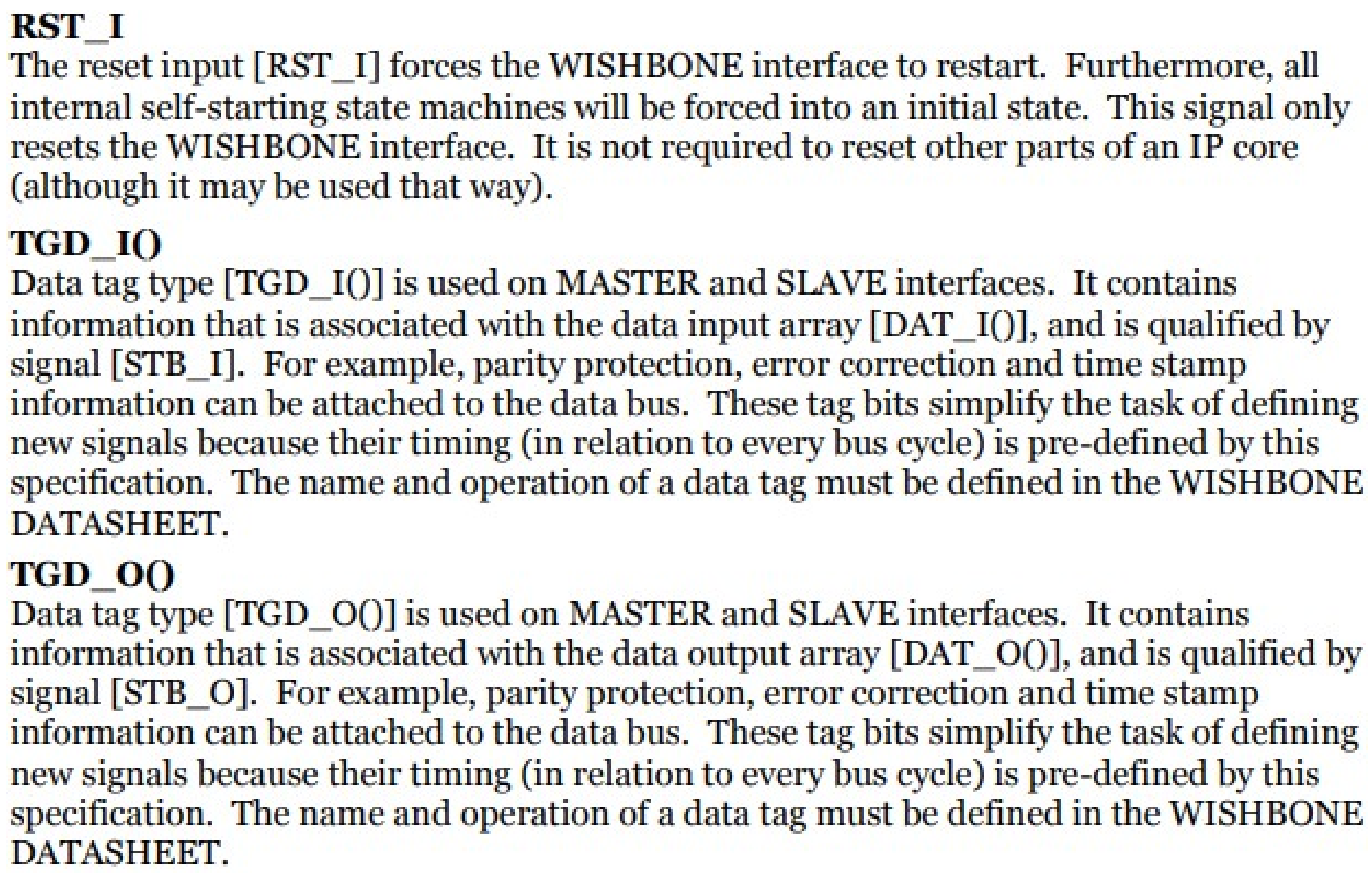


CLK_I
The clock input [CLK_I] coordinates all activities for the internal logic within the WISHBONE interconnect. All WISHBONE output signals are registered at the rising edge of [CLK_I]. All WISHBONE input signals are stable before the rising edge of [CLK_I].

DAT_I()
The data input array [DAT_I()] is used to pass binary data. The array boundaries are determined by the port size, with a maximum port size of 64-bits (e.g. [DAT_I(63..0)]). Also see the [DAT_O()] and [SEL_O()] signal descriptions.

DAT_O()
The data output array [DAT_O()] is used to pass binary data. The array boundaries are determined by the port size, with a maximum port size of 64-bits (e.g. [DAT_I(63..0)]). Also see the [DAT_I()] and [SEL_O()] signal descriptions.

03.3. Інтерфейсні сигнали Wishbone. Signals Common to MASTER and SLAVE Interfaces

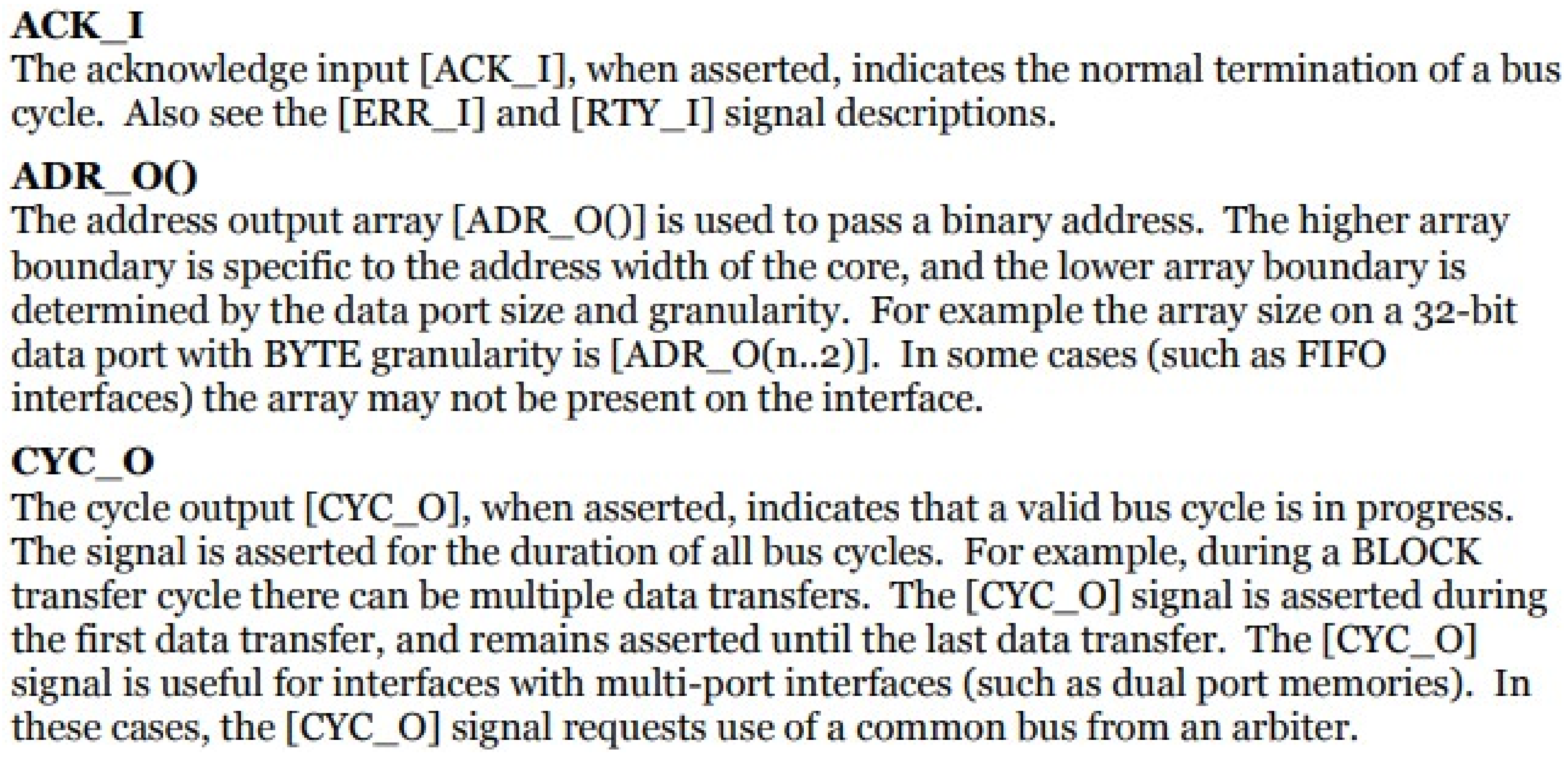


RST_I
The reset input [RST_I] forces the WISHBONE interface to restart. Furthermore, all internal self-starting state machines will be forced into an initial state. This signal only resets the WISHBONE interface. It is not required to reset other parts of an IP core (although it may be used that way).

TGD_I()
Data tag type [TGD_I()] is used on MASTER and SLAVE interfaces. It contains information that is associated with the data input array [DAT_I()], and is qualified by signal [STB_I]. For example, parity protection, error correction and time stamp information can be attached to the data bus. These tag bits simplify the task of defining new signals because their timing (in relation to every bus cycle) is pre-defined by this specification. The name and operation of a data tag must be defined in the WISHBONE DATASHEET.

TGD_O()
Data tag type [TGD_O()] is used on MASTER and SLAVE interfaces. It contains information that is associated with the data output array [DAT_O()], and is qualified by signal [STB_O]. For example, parity protection, error correction and time stamp information can be attached to the data bus. These tag bits simplify the task of defining new signals because their timing (in relation to every bus cycle) is pre-defined by this specification. The name and operation of a data tag must be defined in the WISHBONE DATASHEET.

03.4. Інтерфейсні сигнали Wishbone. MASTER Signals

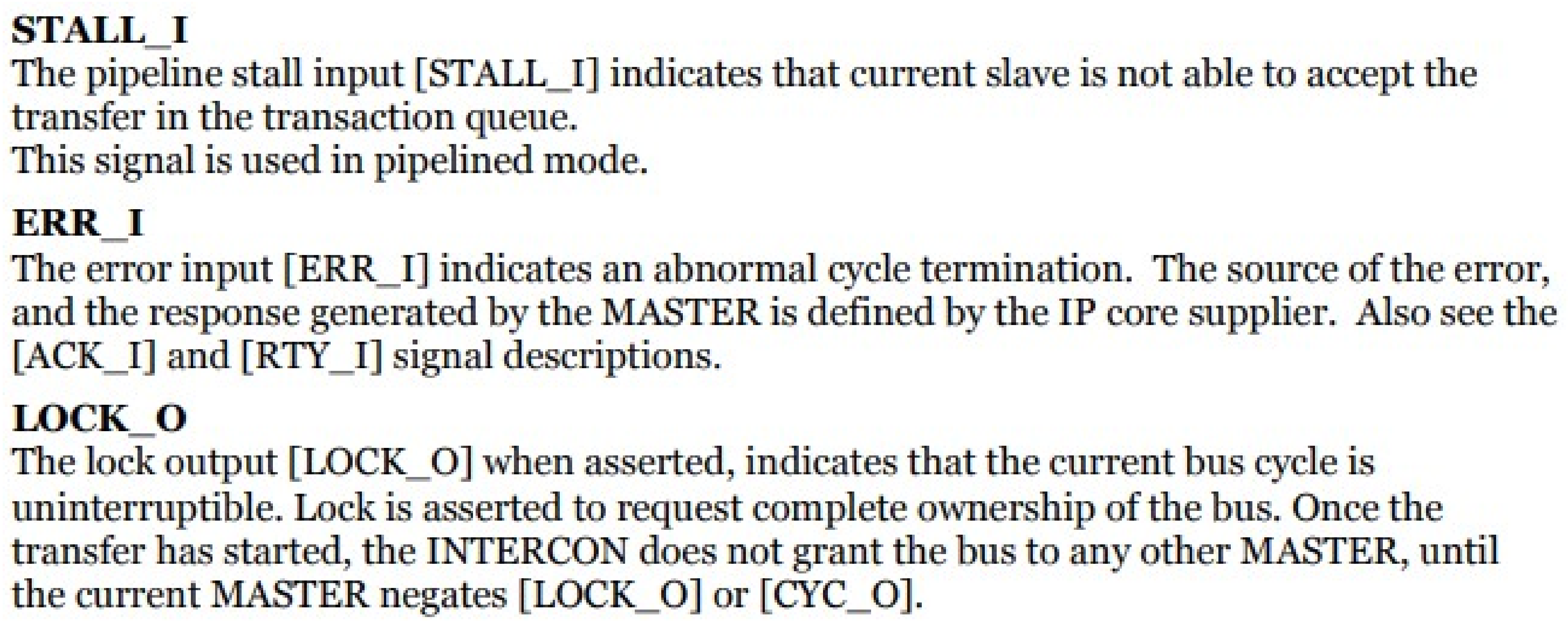


ACK_I
The acknowledge input [ACK_I], when asserted, indicates the normal termination of a bus cycle. Also see the [ERR_I] and [RTY_I] signal descriptions.

ADR_O()
The address output array [ADR_O()] is used to pass a binary address. The higher array boundary is specific to the address width of the core, and the lower array boundary is determined by the data port size and granularity. For example the array size on a 32-bit data port with BYTE granularity is [ADR_O(n..2)]. In some cases (such as FIFO interfaces) the array may not be present on the interface.

CYC_O
The cycle output [CYC_O], when asserted, indicates that a valid bus cycle is in progress. The signal is asserted for the duration of all bus cycles. For example, during a BLOCK transfer cycle there can be multiple data transfers. The [CYC_O] signal is asserted during the first data transfer, and remains asserted until the last data transfer. The [CYC_O] signal is useful for interfaces with multi-port interfaces (such as dual port memories). In these cases, the [CYC_O] signal requests use of a common bus from an arbiter.

03.5. Інтерфейсні сигнали Wishbone. MASTER Signals

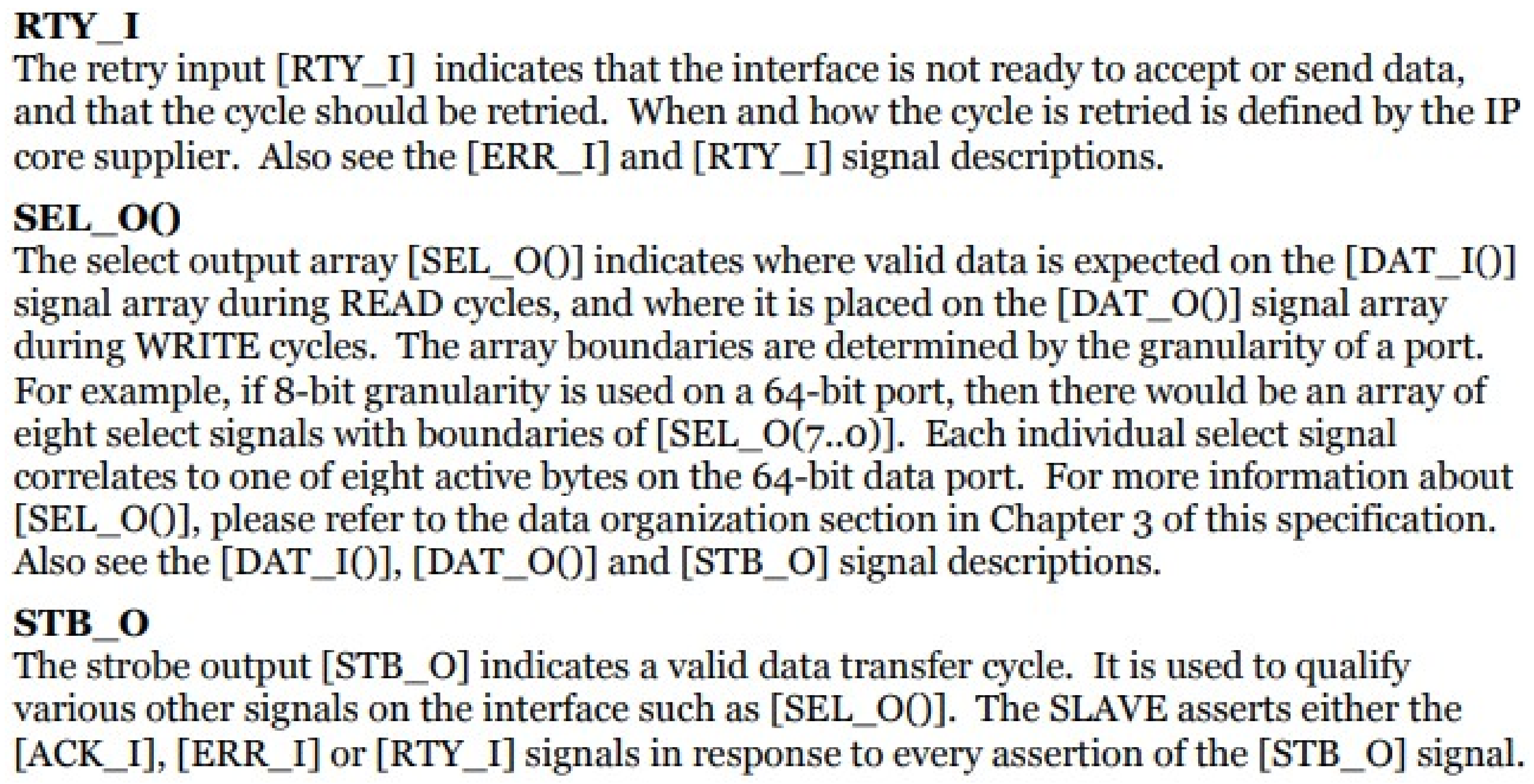


STALL_I
The pipeline stall input [STALL_I] indicates that current slave is not able to accept the transfer in the transaction queue.
This signal is used in pipelined mode.

ERR_I
The error input [ERR_I] indicates an abnormal cycle termination. The source of the error, and the response generated by the MASTER is defined by the IP core supplier. Also see the [ACK_I] and [RTY_I] signal descriptions.

LOCK_O
The lock output [LOCK_O] when asserted, indicates that the current bus cycle is uninterruptible. Lock is asserted to request complete ownership of the bus. Once the transfer has started, the INTERCON does not grant the bus to any other MASTER, until the current MASTER negates [LOCK_O] or [CYC_O].

03.6. Інтерфейсні сигнали Wishbone. MASTER Signals

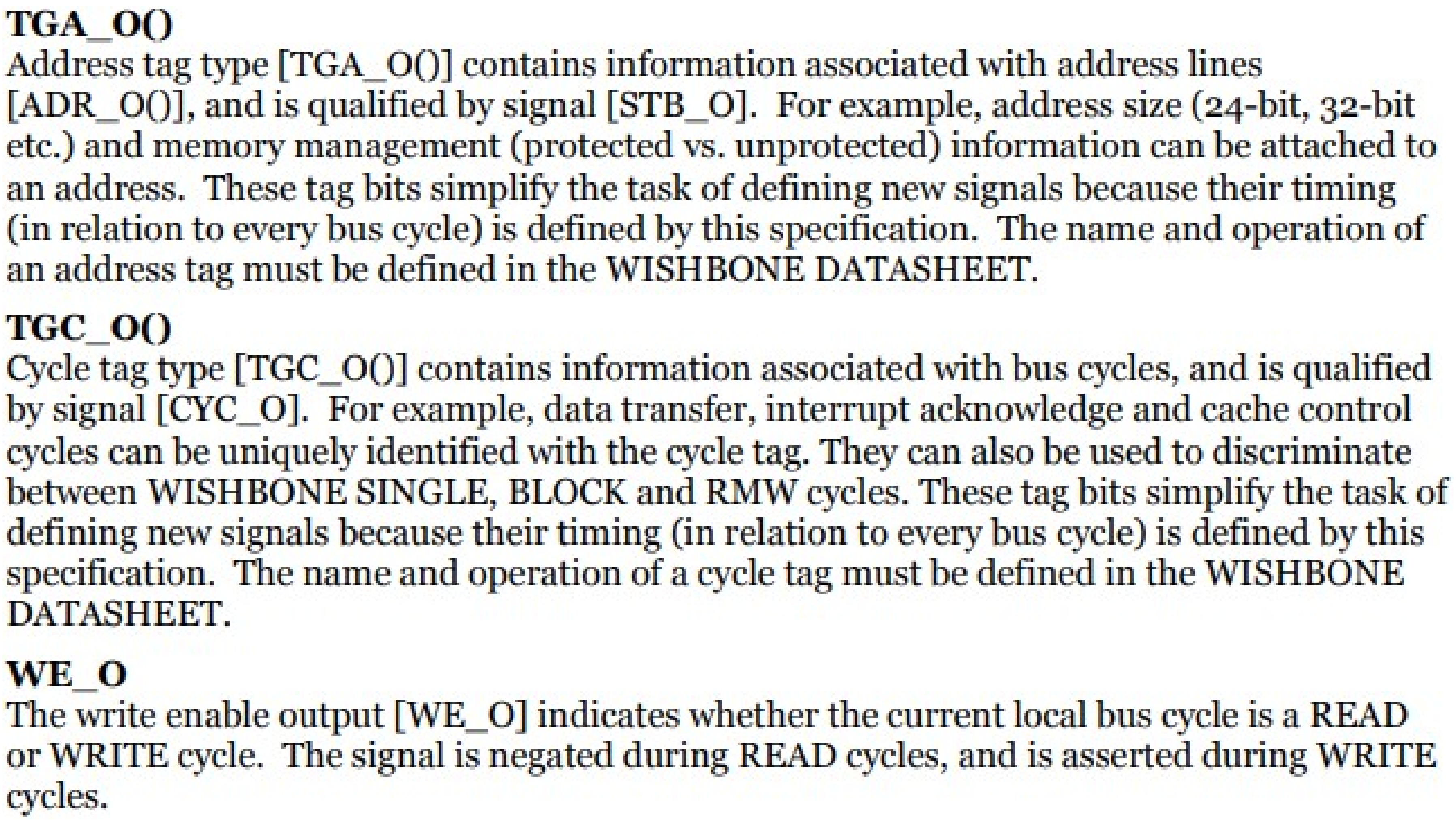


RTY_I
The retry input [RTY_I] indicates that the interface is not ready to accept or send data, and that the cycle should be retried. When and how the cycle is retried is defined by the IP core supplier. Also see the [ERR_I] and [RTY_I] signal descriptions.

SEL_O()
The select output array [SEL_O()] indicates where valid data is expected on the [DAT_I()] signal array during READ cycles, and where it is placed on the [DAT_O()] signal array during WRITE cycles. The array boundaries are determined by the granularity of a port. For example, if 8-bit granularity is used on a 64-bit port, then there would be an array of eight select signals with boundaries of [SEL_O(7..0)]. Each individual select signal correlates to one of eight active bytes on the 64-bit data port. For more information about [SEL_O()], please refer to the data organization section in Chapter 3 of this specification. Also see the [DAT_I()], [DAT_O()] and [STB_O] signal descriptions.

STB_O
The strobe output [STB_O] indicates a valid data transfer cycle. It is used to qualify various other signals on the interface such as [SEL_O()]. The SLAVE asserts either the [ACK_I], [ERR_I] or [RTY_I] signals in response to every assertion of the [STB_O] signal.

03.7. Інтерфейсні сигнали Wishbone. MASTER Signals

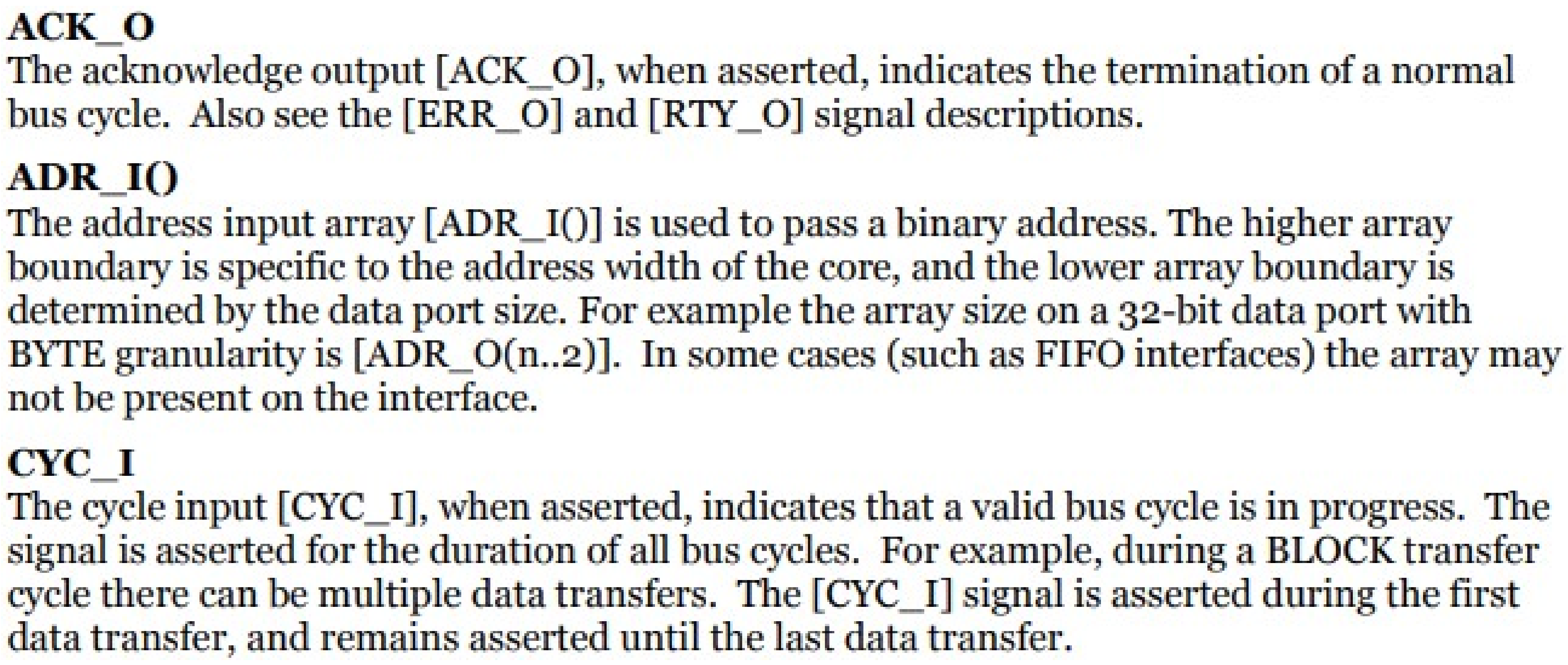


TGA_O()
Address tag type [TGA_O()] contains information associated with address lines [ADR_O()], and is qualified by signal [STB_O]. For example, address size (24-bit, 32-bit etc.) and memory management (protected vs. unprotected) information can be attached to an address. These tag bits simplify the task of defining new signals because their timing (in relation to every bus cycle) is defined by this specification. The name and operation of an address tag must be defined in the WISHBONE DATASHEET.

TGC_O()
Cycle tag type [TGC_O()] contains information associated with bus cycles, and is qualified by signal [CYC_O]. For example, data transfer, interrupt acknowledge and cache control cycles can be uniquely identified with the cycle tag. They can also be used to discriminate between WISHBONE SINGLE, BLOCK and RMW cycles. These tag bits simplify the task of defining new signals because their timing (in relation to every bus cycle) is defined by this specification. The name and operation of a cycle tag must be defined in the WISHBONE DATASHEET.

WE_O
The write enable output [WE_O] indicates whether the current local bus cycle is a READ or WRITE cycle. The signal is negated during READ cycles, and is asserted during WRITE cycles.

03.8. Інтерфейсні сигнали Wishbone. SLAVE Signals

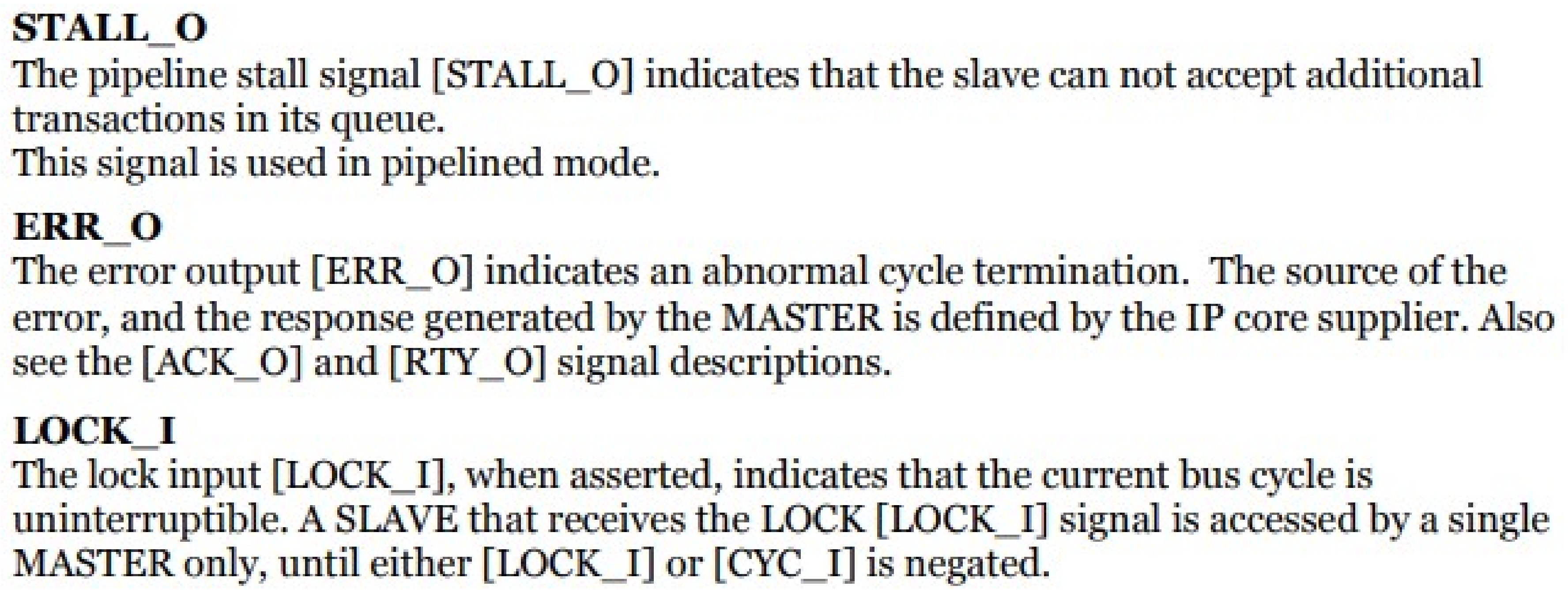


ACK_O
The acknowledge output [ACK_O], when asserted, indicates the termination of a normal bus cycle. Also see the [ERR_O] and [RTY_O] signal descriptions.

ADR_I()
The address input array [ADR_I()] is used to pass a binary address. The higher array boundary is specific to the address width of the core, and the lower array boundary is determined by the data port size. For example the array size on a 32-bit data port with BYTE granularity is [ADR_O(n..2)]. In some cases (such as FIFO interfaces) the array may not be present on the interface.

CYC_I
The cycle input [CYC_I], when asserted, indicates that a valid bus cycle is in progress. The signal is asserted for the duration of all bus cycles. For example, during a BLOCK transfer cycle there can be multiple data transfers. The [CYC_I] signal is asserted during the first data transfer, and remains asserted until the last data transfer.

03.9. Інтерфейсні сигнали Wishbone. SLAVE Signals

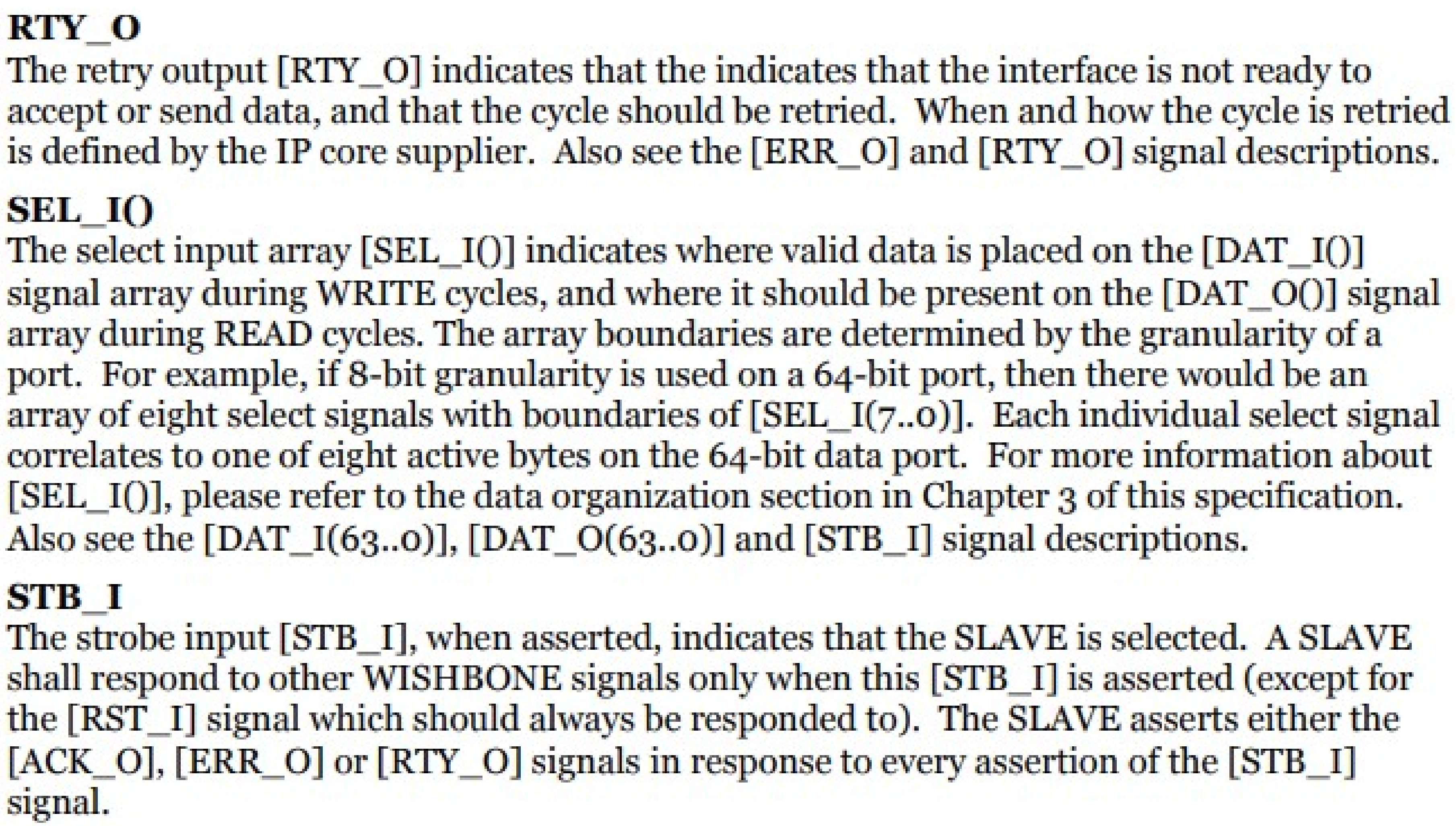


STALL_O
The pipeline stall signal [STALL_O] indicates that the slave can not accept additional transactions in its queue.
This signal is used in pipelined mode.

ERR_O
The error output [ERR_O] indicates an abnormal cycle termination. The source of the error, and the response generated by the MASTER is defined by the IP core supplier. Also see the [ACK_O] and [RTY_O] signal descriptions.

LOCK_I
The lock input [LOCK_I], when asserted, indicates that the current bus cycle is uninterruptible. A SLAVE that receives the LOCK [LOCK_I] signal is accessed by a single MASTER only, until either [LOCK_I] or [CYC_I] is negated.

03.10. Інтерфейсні сигнали Wishbone. SLAVE Signals

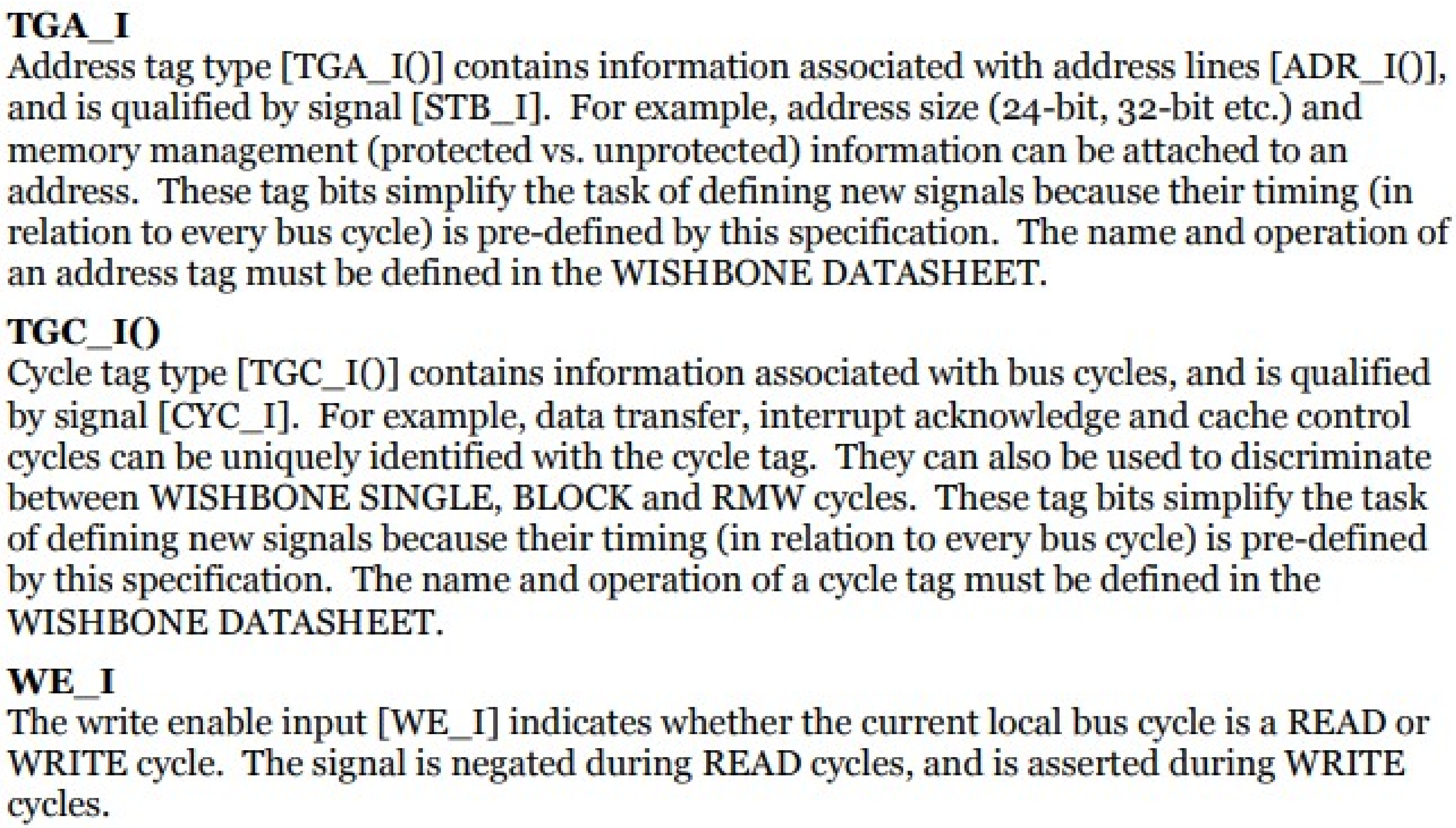


RTY_O
The retry output [RTY_O] indicates that the indicates that the interface is not ready to accept or send data, and that the cycle should be retried. When and how the cycle is retried is defined by the IP core supplier. Also see the [ERR_O] and [RTY_O] signal descriptions.

SEL_I()
The select input array [SEL_I()] indicates where valid data is placed on the [DAT_I()] signal array during WRITE cycles, and where it should be present on the [DAT_O()] signal array during READ cycles. The array boundaries are determined by the granularity of a port. For example, if 8-bit granularity is used on a 64-bit port, then there would be an array of eight select signals with boundaries of [SEL_I(7..0)]. Each individual select signal correlates to one of eight active bytes on the 64-bit data port. For more information about [SEL_I()], please refer to the data organization section in Chapter 3 of this specification. Also see the [DAT_I(63..0)], [DAT_O(63..0)] and [STB_I] signal descriptions.

STB_I
The strobe input [STB_I], when asserted, indicates that the SLAVE is selected. A SLAVE shall respond to other WISHBONE signals only when this [STB_I] is asserted (except for the [RST_I] signal which should always be responded to). The SLAVE asserts either the [ACK_O], [ERR_O] or [RTY_O] signals in response to every assertion of the [STB_I] signal.

03.11. Інтерфейсні сигнали Wishbone. SLAVE Signals



TGA_I
Address tag type [TGA_I()] contains information associated with address lines [ADR_I()], and is qualified by signal [STB_I]. For example, address size (24-bit, 32-bit etc.) and memory management (protected vs. unprotected) information can be attached to an address. These tag bits simplify the task of defining new signals because their timing (in relation to every bus cycle) is pre-defined by this specification. The name and operation of an address tag must be defined in the WISHBONE DATASHEET.

TGC_I()
Cycle tag type [TGC_I()] contains information associated with bus cycles, and is qualified by signal [CYC_I]. For example, data transfer, interrupt acknowledge and cache control cycles can be uniquely identified with the cycle tag. They can also be used to discriminate between WISHBONE SINGLE, BLOCK and RMW cycles. These tag bits simplify the task of defining new signals because their timing (in relation to every bus cycle) is pre-defined by this specification. The name and operation of a cycle tag must be defined in the WISHBONE DATASHEET.

WE_I
The write enable input [WE_I] indicates whether the current local bus cycle is a READ or WRITE cycle. The signal is negated during READ cycles, and is asserted during WRITE cycles.

04.1. Цикли шини WISHBONE.

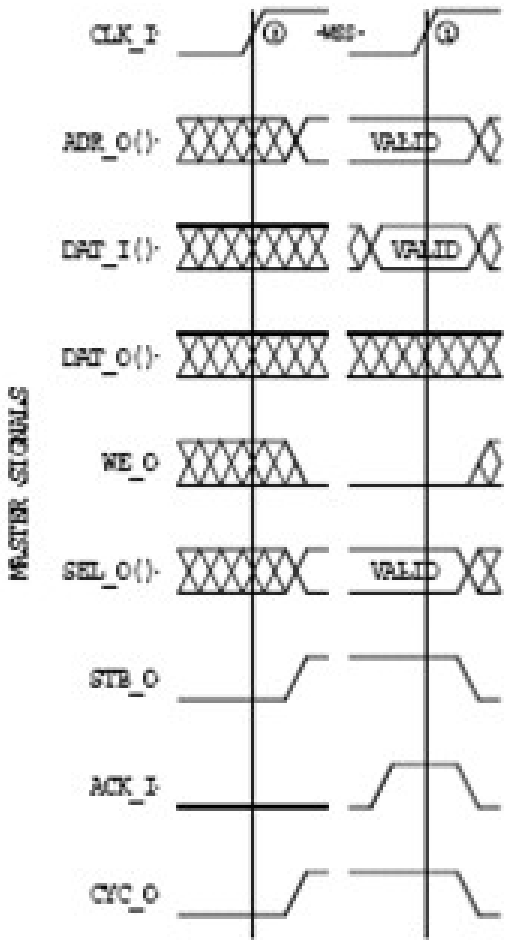
There are three types of defined WISHBONE bus cycles. They include:

- SINGLE READ/WRITE (одиничний цикл читання/запис);
- BLOCK READ/WRITE (блокове читання/запис, множинні читання/записи);
- READ MODIFY WRITE (RMW) (читання/модифікація/запис).

04.2. Цикли шини WISHBONE.

SINGLE READ/WRITE

1. In response to clock edge 0, the MASTER interface asserts [ADR_O()], [WE_O], [SEL_O], [STB_O] and [CYC_O].
2. The SLAVE decodes the bus cycle by monitoring its [STB_I] and address inputs, and presents valid data on its [DAT_O()] lines. Because the system is in a point-to-point configuration, the SLAVE [DAT_O()] signals are connected to the MASTER [DAT_I()] signals.
3. The SLAVE indicates that it has placed valid data on the data bus by asserting the MASTER's [ACK_I] acknowledge signal. Also note that the SLAVE may delay its response by inserting one or more wait states. In this case, the SLAVE does not assert the acknowledge line. The possibility of a wait state in the timing diagrams is indicated by '-WSS-'.
4. The MASTER monitors the state of its [ACK_I] line, and determines that the SLAVE has acknowledged the transfer at clock edge 1.
5. The MASTER latches [DAT_I()] and negates its [STB_O] signal in response to [ACK_I].



04.3. Цикли шини WISHBONE.

BLOCK READ/WRITE

The BLOCK READ/WRITE cycles are very similar to the SINGLE READ/WRITE cycles. The BLOCK cycles can be thought of as two or more back-to-back SINGLE cycles strung together. In WISHBONE terminology the term cycle refers to the whole BLOCK cycle. The small, individual data transfers that make up the BLOCK cycle are called phases.

The starting and stopping point of the BLOCK cycles are identified by the assertion and negation of the MASTER [CYC_O] signal (respectively). The [CYC_O] signal is also used in shared bus and crossbar interconnections because it informs system logic that the MASTER wishes to use the bus. It also informs the interconnection when it is through with its bus cycle.

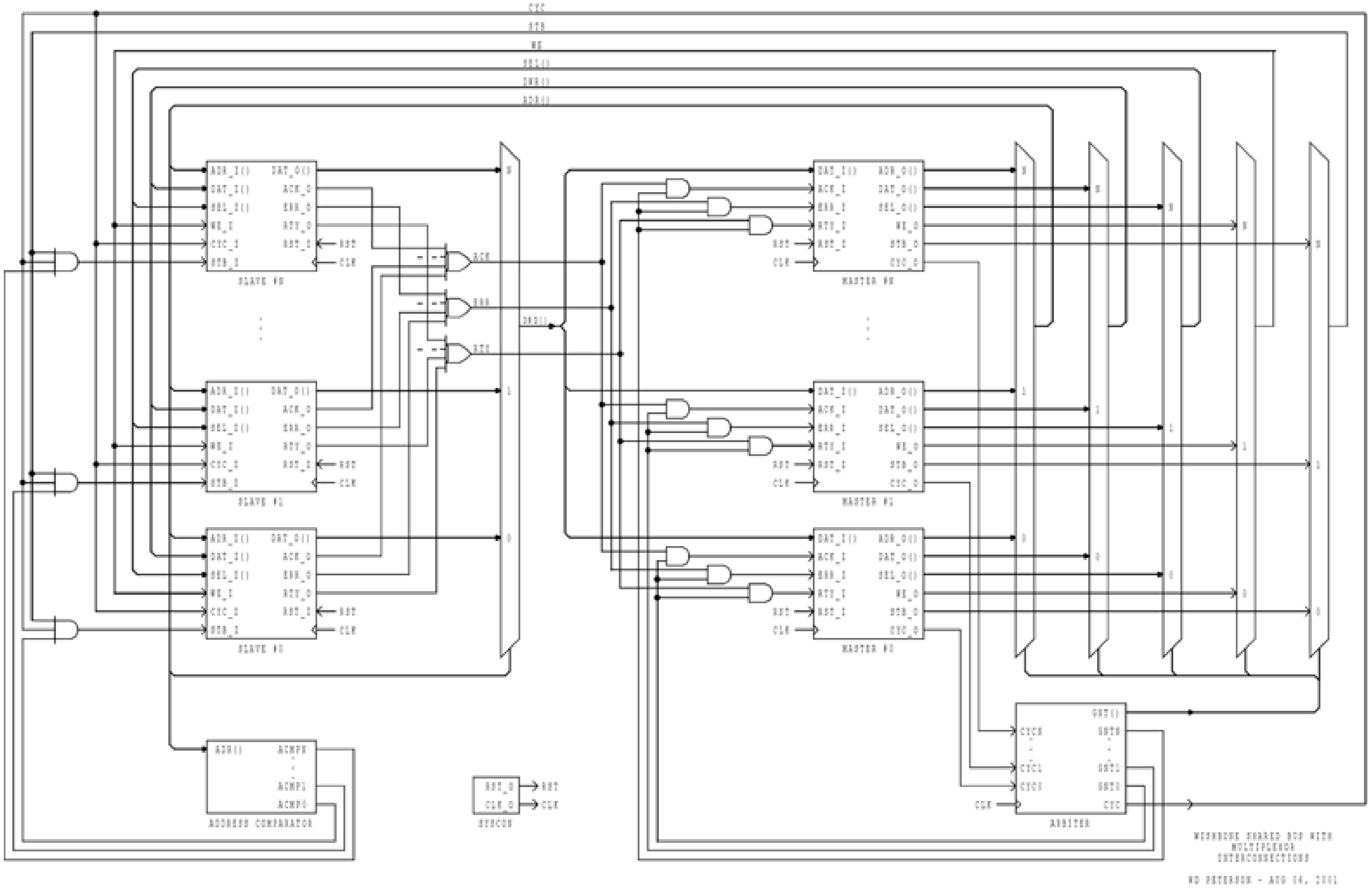
04.4. Цикли шини WISHBONE.

READ MODIFY WRITE (RMW)

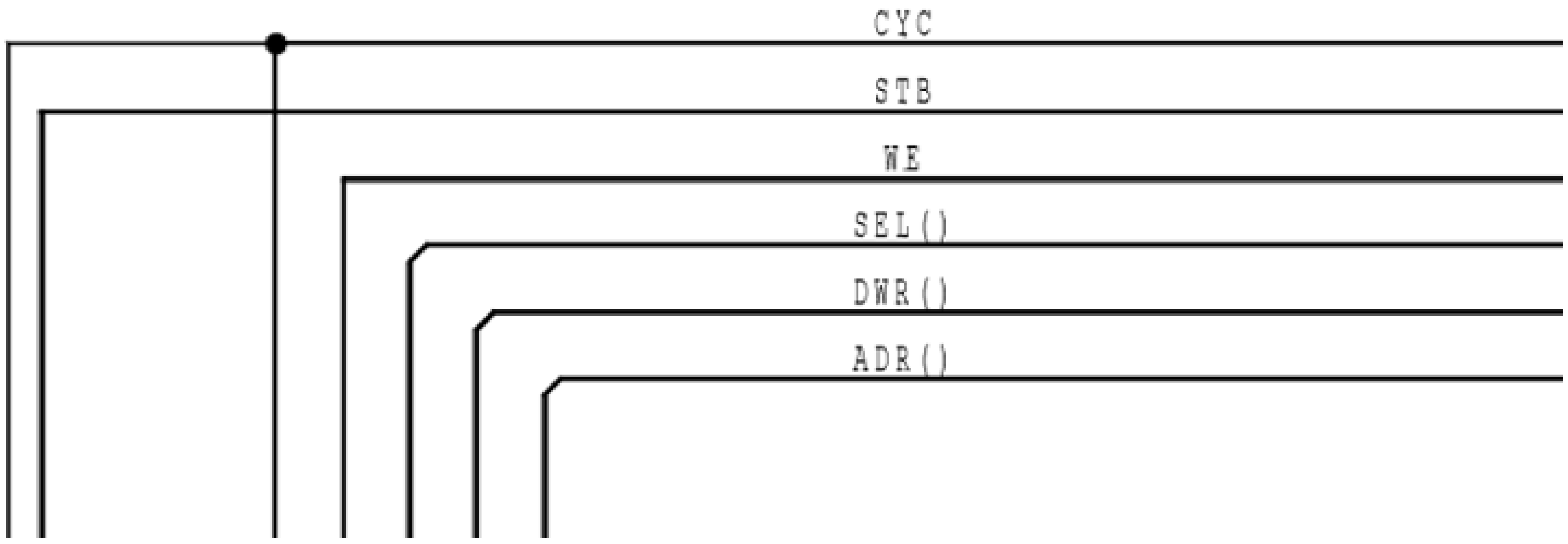
Цикл READ-MODIFY-WRITE використовується у багатопроцесорних та багатозадачних системах. Цей спеціальний цикл дозволяє декільком програмним процесам ділитися спільними ресурсами за допомогою семафорів. Зазвичай це робиться на інтерфейсах для дискових контролерів, послідовних портів і пам'яті. Як зрозуміло з назви, цикл READ-MODIFY-WRITE зчитує та записує дані до місця пам’яті за один цикл шини. Це запобігає розподілу загального ресурсу для двох або більше процесів. Цикл READ-MODIFY-WRITE також можна розглядати як неподільний цикл.

Цикл READ-MODIFY-WRITE також відомий як неподільний цикл. Важливо відзначити, що процесор (або інший пристрій, підключений до інтерфейсу MASTER) повинен підтримувати цикл RMW. Як правило, це робиться за допомогою спеціальних інструкцій порівняння та встановлення (CAS) чи перевірки та встановлення (TAS) .

05.1. Арбітр шини WISHBONE.

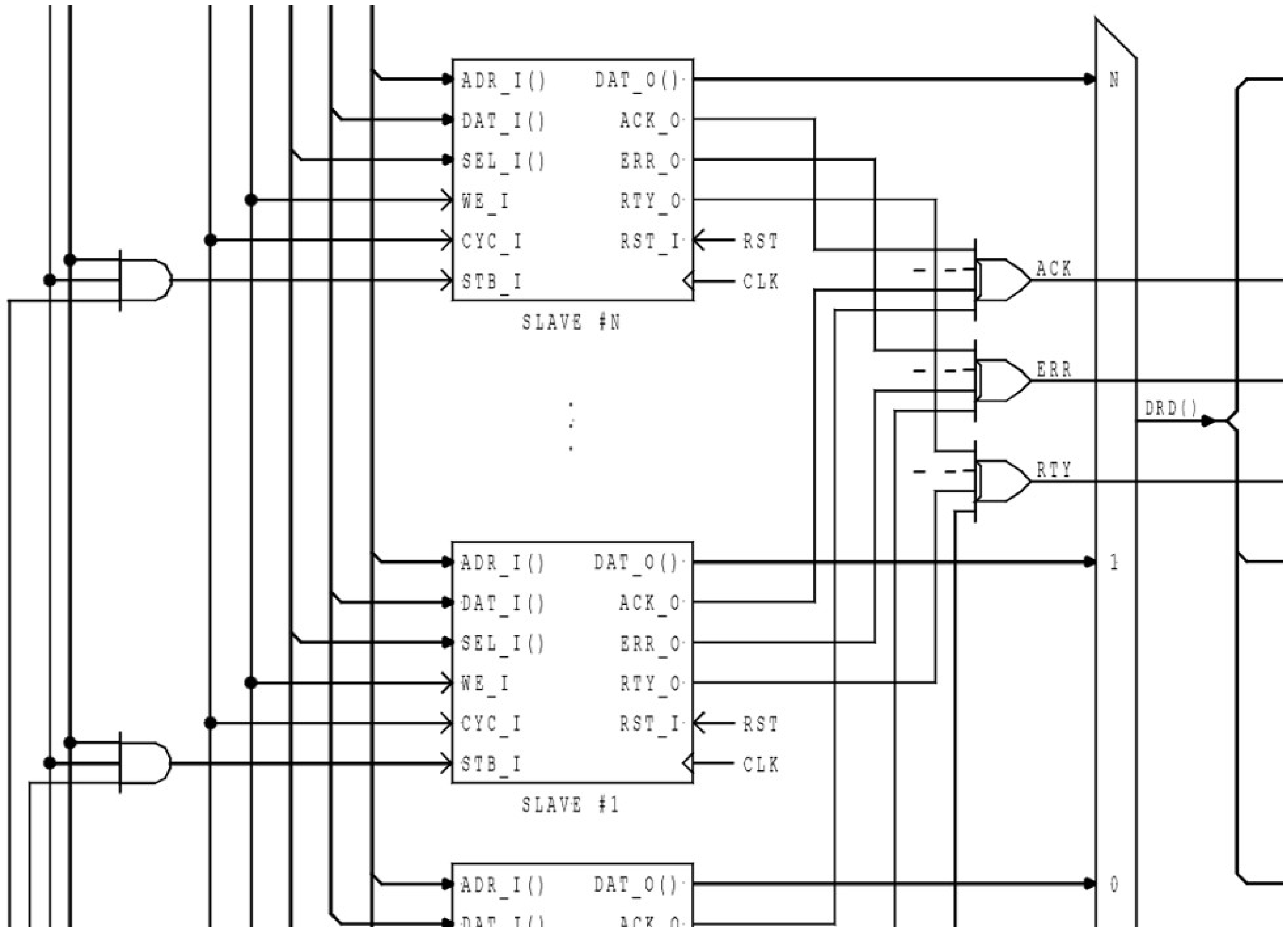


05.2. Арбітр шини WISHBONE.

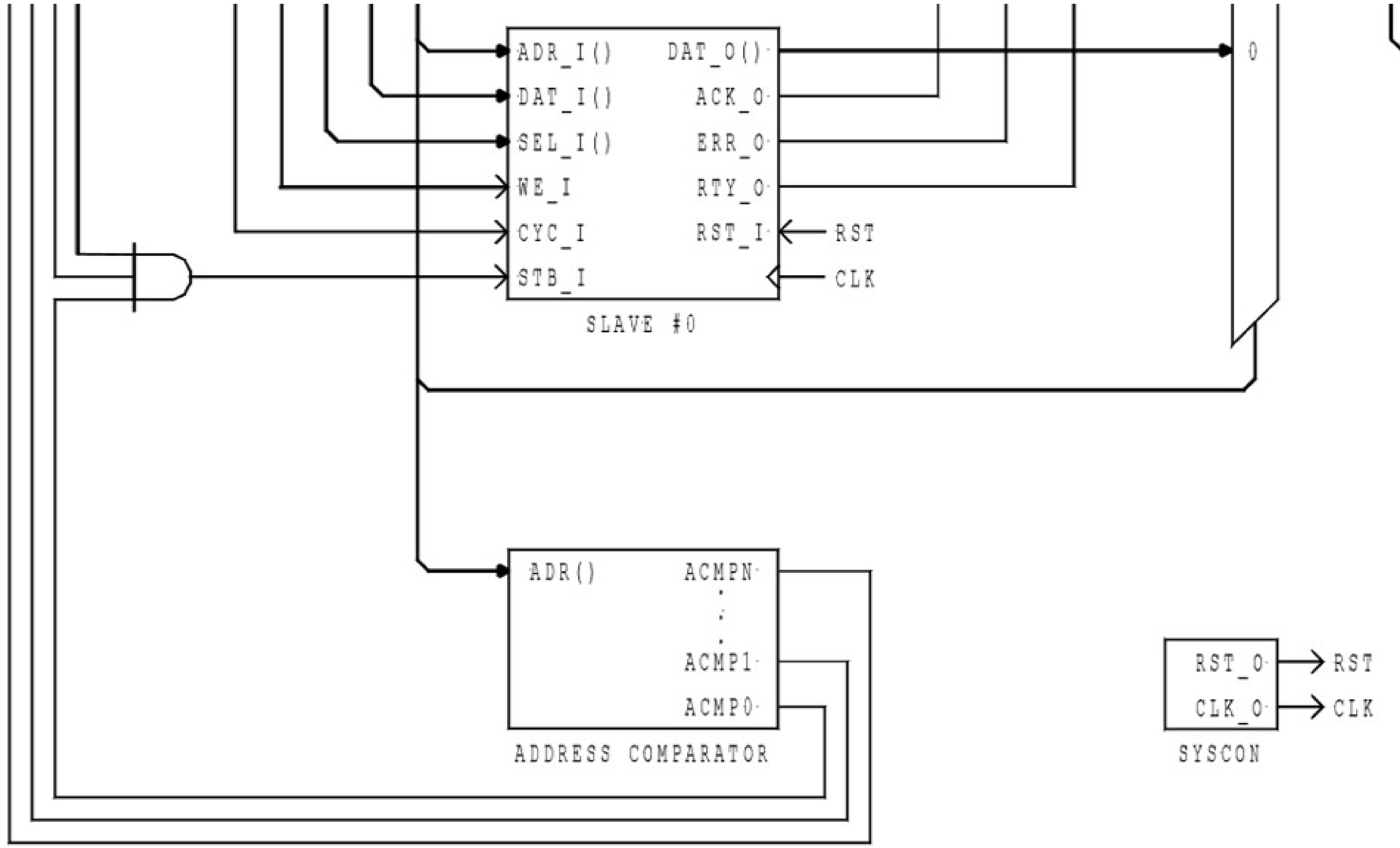


05.3. Арбітр шини WISHBONE.

–

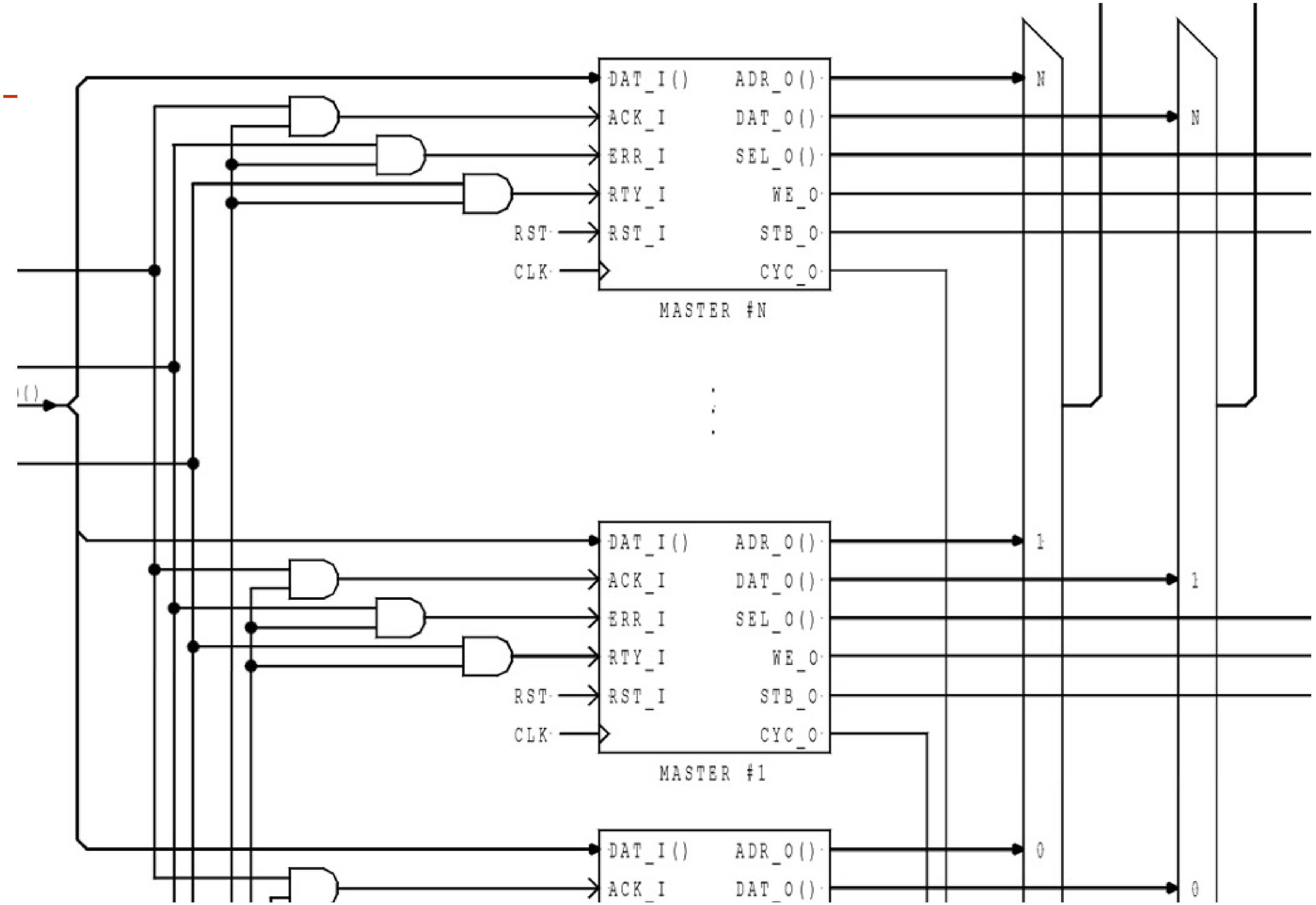


05.4. Арбітр шини WISHBONE.

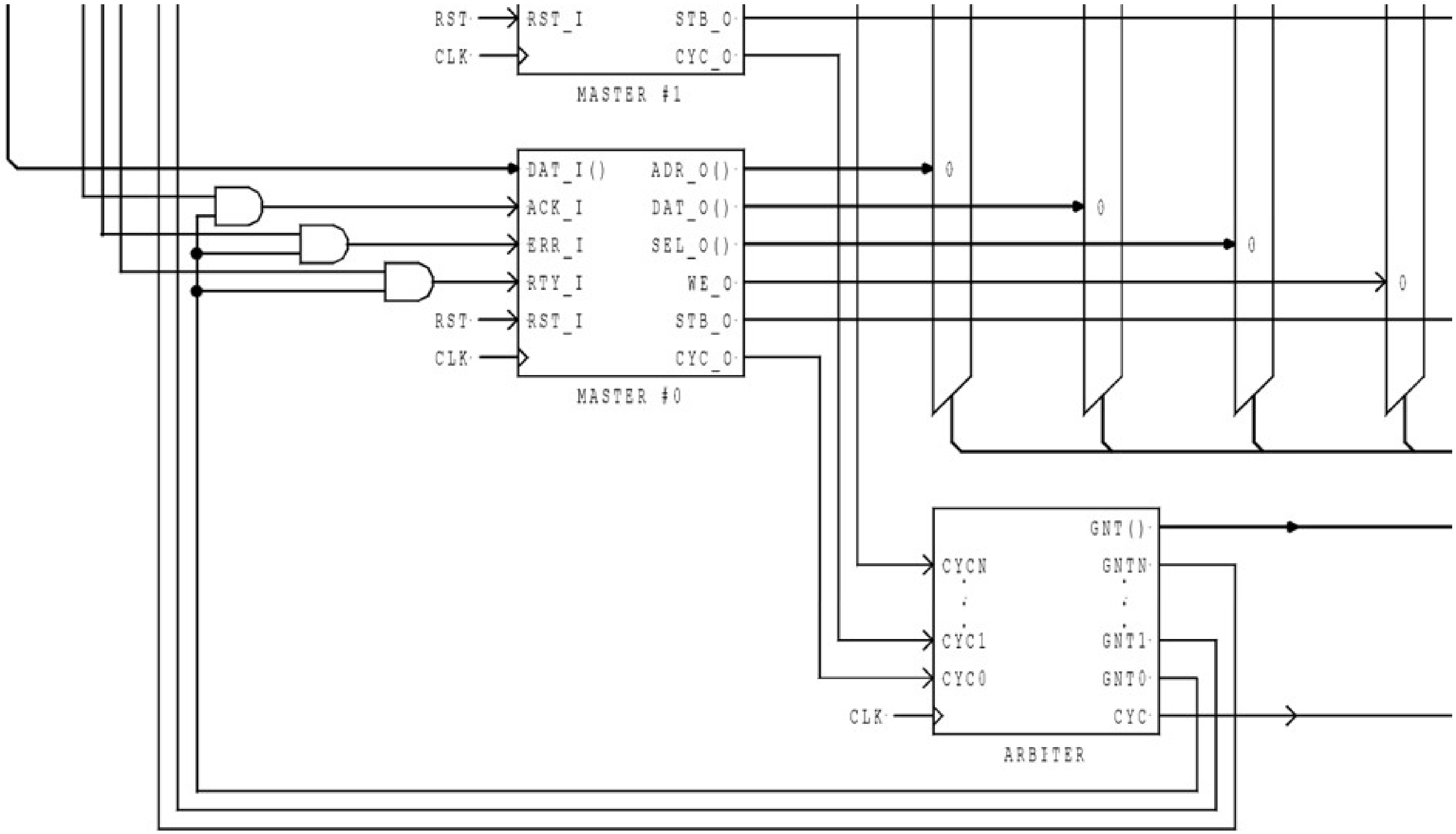


05.5. Арбітр шини WISHBONE.

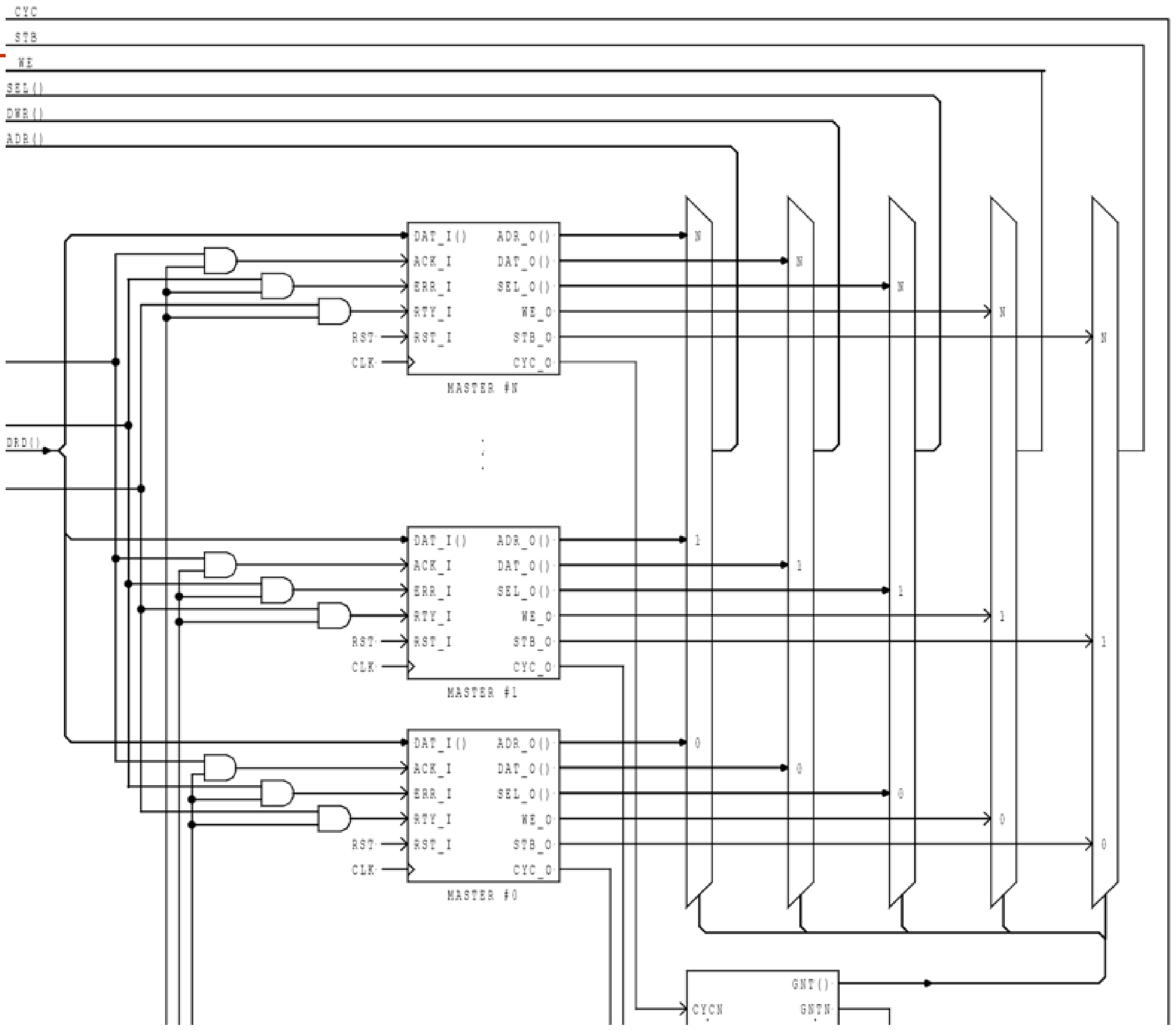
–



05.6. Арбітр шини WISHBONE.

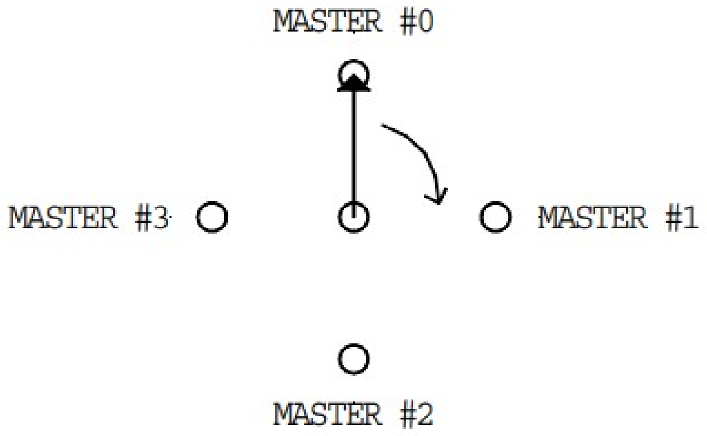


05.7. Арбітр шини WISHBONE.

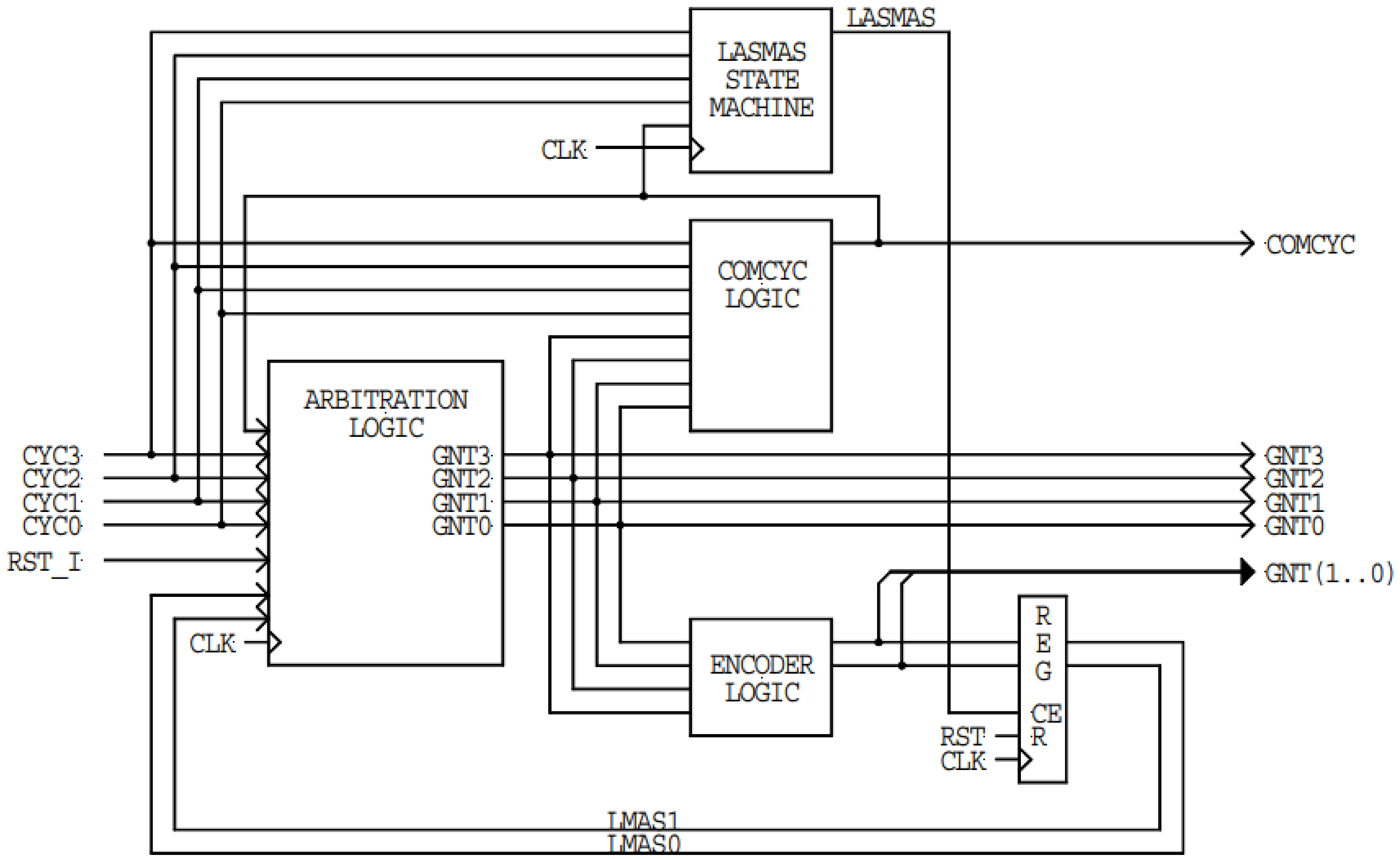


05.8. Арбітр шини WISHBONE. *(4 Level, Round-robin Arbiter)*

Round-robin arbiters give equal priority to all of the MASTERs. They grant the bus on a rotating basis much like the four position rotary switch shown in Figure . When a MASTER relinquishes the bus, the switch is turned to the next position and the bus is granted to the MASTER on that level. If a MASTER on any particular level is not making a bus request, then the arbiter skips over that level and continues onto the next one. In this way all of the MASTERs are granted the bus on an equal basis.



05.9. Арбітр шини WISHBONE.
(4 Level, Round-robin Arbiter)



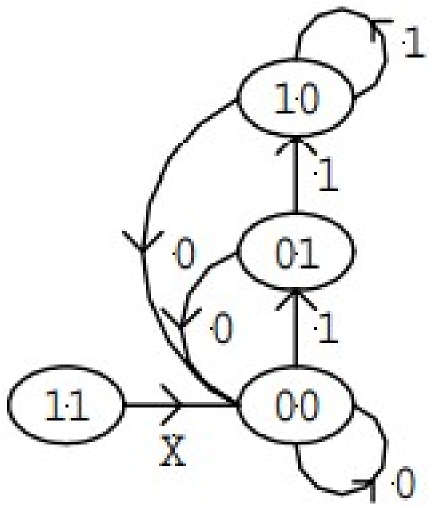
05.10. Арбітр шини WISHBONE.
(4 Level, Round-robin Arbiter)

```
ARBITRATION LOGIC (REGISTERED OUTPUT):

GNT0 := ( /RST * /COMCYC * /LMAS1 * /LMAS0 * /CYC3 * /CYC2 * /CYC1 *  CYC0 )
      + ( /RST * /COMCYC * /LMAS1 *  LMAS0 * /CYC3 * /CYC2         *  CYC0 )
      + ( /RST * /COMCYC *  LMAS1 * /LMAS0 * /CYC3                 *  CYC0 )
      + ( /RST * /COMCYC *  LMAS1 *  LMAS0                         *  CYC0 )
      + ( /RST *  COMCYC *  GNT0 );

GNT1 := ( /RST * /COMCYC * /LMAS1 * /LMAS0                 *  CYC1         )
      + ( /RST * /COMCYC * /LMAS1 *  LMAS0 * /CYC3 * /CYC2 *  CYC1 * /CYC0 )
      + ( /RST * /COMCYC *  LMAS1 * /LMAS0 * /CYC3         *  CYC1 * /CYC0 )
      + ( /RST * /COMCYC *  LMAS1 *  LMAS0                 *  CYC1   /CYC0 )
      + ( /RST *  COMCYC *  GNT1 );

GNT2 := ( /RST * /COMCYC * /LMAS1 * /LMAS0           CYC2 * /CYC1         )
      + ( /RST * /COMCYC * /LMAS1 *  LMAS0           CYC2                 )
      + ( /RST * /COMCYC *  LMAS1 * /LMAS0 * /CYC3 * CYC2 * /CYC1 * /CYC0 )
      + ( /RST * /COMCYC *  LMAS1 *  LMAS0           CYC2 * /CYC1 * /CYC0 )
      + ( /RST *  COMCYC *  GNT2 );

GNT3 := ( /RST * /COMCYC * /LMAS1 * /LMAS0 *  CYC3 * /CYC2 * /CYC1         )
      + ( /RST * /COMCYC * /LMAS1 *  LMAS0 *  CYC3 * /CYC2                 )
      + ( /RST * /COMCYC *  LMAS1 * /LMAS0 *  CYC3                         )
      + ( /RST * /COMCYC *  LMAS1 *  LMAS0 *  CYC3 * /CYC2 * /CYC1 * /CYC0 )
      + ( /RST *  COMCYC *  GNT3 );
```

```
COMCYC LOGIC:

COMCYC = ( CYC3 * GNT3 )
       + ( CYC2 * GNT2 )
       + ( CYC1 * GNT1 )
       + ( CYC0 * GNT0 );

ENCODER LOGIC:

GNT(1) = GNT3 + GNT2;
GNT(0) = GNT3 + GNT1;

SHORTHAND NOTATION:

 =    COMBINATORIAL OUTPUT
:=    REGISTERED OUTPUT
 +    LOGICAL 'OR'
 *    LOGICAL 'AND'
 /    LOGICAL 'NOT'
```

05.11. Арбітр шини WISHBONE.
(4 Level, Round-robin Arbiter)

```
'LASMAS' STATE MACHINE:

GENERATES A CLOCK ENABLE TO THE LAST MASTER
REGISTER.

INPUTS: BEGIN
STATES: EDGE, LASMAS

RESET:  ALL 'CYC' INPUTS FORCED LOW, THEREBY
        RESETTING THE ARBITER.


STATE MACHINE EQUATIONS:

BEG     = (CYC0 + CYC1 + CYC2 + CYC3) * /COMCYC;

LASMAS := ( BEG * /EDGE * /LASMAS );

EDGE   := ( BEG * /EDGE *  LASMAS )
        + ( BEG *  EDGE * /LASMAS );
```



STATE DIAGRAM