RZ 2931 (# 91526) 06/09/97
Engineering Technology 38 pages

Research Report

SMART GPPINT Specification
Version 1.5

P. Buchmann

IBM Research Division
Zurich Research Laboratory
8803 Rüschlikon
Switzerland

LIMITED DISTRIBUTION NOTICE
This report has been submitted for publication outside of IBM and will probably be copyrighted if accepted for publication. It has been issued as a Research Report for early dissemination of its contents and will be distributed outside of IBM up to one year after the date indicated at the top of this page. In view of transfer of copyright to the outside publisher, its distribution outside of IBM prior to publication should be limited to peer communications and to specific requests. After outside publication, requests should be filled only by reprints or legally obtained copies of the article (e.g., payment of royalties).

IBM Research Division
Almaden, T.J. Watson, Tokyo, Zurich

SMART GPPINT Specification

P. Buchmann

IBM Research Division, Zurich Research Laboratory, 8803 Rüschlikon, Switzerland

ABSTRACT: After a short introduction on SMART architecture this report describes the General Purpose Processor Interface module (GPPINT) of the SMART chip. This functional unit provides an interface (synchronous or asynchronous) to an external microprocessor and several internal interfaces to other modules of SMART. These modules operate with clocks that are determined by the actual configuration of the chip. Therefore, all internal interfaces operate in asynchronous mode. Global configuration information is written into the GPPINT by the microprocessor; module-specific configuration data is forwarded by the GPPINT to the modules. Status and counter information is read out from the modules via GPPINT. Other features of the GPPINT include individual module resetting, clock monitoring and interrupt handling.

SMART

GPPINT Specification

Version 1.5

11-26-1996

This document describes the
General Purpose Processor Interface (GPPINT) module
of the SMART chip

Peter Buchmann

IBM Zurich Research Laboratory

Saeumerstrasse 4
CH-8803 Rueschlikon

Tel.: +41-1-7248 214
FAX: +41-1-710 36 08
email: PBU @ zurich.ibm.com

1 Contents

CONTENTS 2
ABOUT THIS DOCUMENT 3
General 3
History 3
IBM NOTATION 4
ABBREVIATIONS 5
INTRODUCTION 6
OVERVIEW 9
GPPINT, Chiplets and Clock Islands 9
APS: Automatic Protection Switching 13
GPPINT CHIP INTERNAL INTERFACES 15
Functional Description and Timing 15
Chiplet Reset 19
Clock Monitoring 21
Interrupts 22
Statistic and Error Counters 24
Configuration, Command and Status Registers 27
Internal Interface Ports 27
CHIPLET ADDRESS MAPPING 34
EXTERNAL MICROPROCESSOR INTERFACE 35
REFERENCES 38

2 About This Document

2.1 General

This document describes the various interfaces of the General Purpose Processor Interface module (GPPINT) of the SMART[1] chip. GPPINT has two types of interfaces: one chip-external and several chip-internal interfaces. The chip-internal interfaces connect to all other functional modules of the chip. The chip-external interface is designed to fit as much as possible to the MC68360 ([9,10]) and to allow both synchronous and asynchronous mode of operation. A data sheet covering AC/DC characteristics will be added as a separate document[2].

2.2 History

In the following table the document history of distributed versions is listed.

Version	Date	Comments
1.0	8/24/95	First distributed version.
1.1	11-15-95	Second distributed version. Static and non-static configuration and status information of the individual chiplets moved to a separate document. Major change in external GPPINT interface. Interfacing to MC68360 first priority.
1.2	11-16-1995	Correction of some typos.
1.3	1-30-1996	Interrupt and Fatal Error description and figures updated. Chiplet address space included.
1.4	11-22-1996	Document take over by Peter Buchmann Completely reworked chapter “GPPINT Chip Internal Interfaces”. Removed chapter and paragraphs on Fatal Errors (obsolete). Updated figures: removed Fatal Errors, added reset logic, corrected GPPAddr vector, corrected timing. Adapted to new GPPINT handler design. Updated SMART start-up sequence. Major changes to chapters “Statistic and Error Counters”, “Configuration, Command and Status Registers”, “Internal Interface Ports” and “Chiplet Address Mapping”. Better distinction between “chiplet” and “clock island”.
1.5	11-26-1996	Addition of chapter 9 : “External Microprocessor Interface” . Updated chapter 7.1.

Table 2.1: History of distributed versions

Wolfram Lemppenau, who started this document and has written a good part of it, is greatly acknowledged for his work and the many intensive and helpful discussions we had on various issues.

P. Buchmann

[1] SMART: Scalable and Modular ARchitecture for SONET/SDH Technology.

[2] The AC/DC characteristics of the chip-external microprocessor interface will be available after physical design only.

3 IBM Notation

Standard IBM convention is used:

Standard IBM convention is to number the bits in ascending order from left to right. This notation will be used in this documentation unless otherwise noted. This notation is depicted out for one byte and for a 32bit word (four bytes) in Figure 3.1.

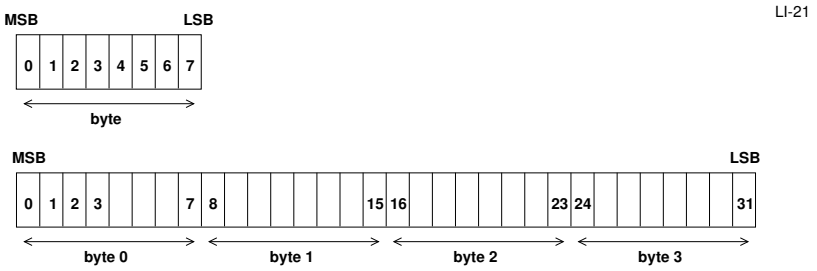

Figure 3.1: IBM notation (example)

Throughout this document, bit0 (Most Significant Bit, MSB) of a byte with name XYZ is always referred to as XYZ[0] and is illustrated at the left in all diagrams unless otherwise noted.

The value of a byte is described in hexadecimal or binary format:

- 5DH represents the decimal value 93D. The suffix D identifies the decimal format. The suffix H identifies the hexadecimal format.
- 5DH is equal to 01011101B. The suffix B identifies the binary format.
- 5DH = 01011101B: bit0 = 0, bit1 = 1, bit2 = 0, bit3 = 1, bit4 = 1, bit5 = 1, bit6 = 0, bit7 = 1 (from left to right, bit0 = MSB, bit7 = LSB, Least Significant Bit).

Furthermore, in the document the terms

- *“xyz = 0”* and “*xyz set low”*
- *“xyz = 1”* and “*xyz set high”*
- *“0” and “low”*
- *“1”* and *“high”*

are equally used.

4 Abbreviations

ACB	ATM Cell Buffer
ACD	ATM Cell Descrambler
ACH	ATM Cell Handler
ACI	ATM Cell Interface
ACS	ATM Cell Scrambler
ADM	Add/Drop Multiplexer
APS	Automatic Protection Switching
ATM	Asynchronous Transfer Mode
AU	Administrative Unit
CS	Circuit Switched
CMP	Clock Monitor Period
DS	Data Strobe
DTACK	Data Transfer Acknowledge
DXC	Digital X(cross) Connect System
FSM	Finite State Machine
GPPINT	General Purpose Processor Interface
GRA	Growable Register Array
HEC	Header Error Check
JTAG	Joint Tag Action Group
ICI	Idle Cell Insertion
ICR	Idle Cell Removal
IRQ	Interrupt Request
ITU	International Telecommunication Union
LI	Line Interface
LIM	Line Interface Macro
LIS	Line Interface Section
LSB	Least Significant Bit
MSB	Most Significant Bit
MSOH	Multiplex Section Overhead
NNI	Network Node Interface
OC	Optical Carrier
OC-1	Optical Carrier level 1
OFP	Overhead and Frame Processor
PDH	Plesiochronous Digital Hierarchy
PECL	Pseudo Emitter Coupled Logic
PECL diff.	Pseudo Emitter Coupled Logic differential
PLL	Phase Locked Loop
POH	Path Overhead
RAM	Random Access Memory
RSOH	Regenerator Section Overhead
SDF	SDH Deframer
SBU	SDH Buffer
SDH	Synchronous Digital Hierarchy
SFD	SDH Frame Descrambler
SFR	SDH Framer
SFS	SDH Frame Scrambler
SOH	Section Overhead SOH = RSOH + MSOH
SMART	Scalable and Modular Architecture for SONET/SDH Technology
SONET	Synchronous Optical Network
SPE	Synchronous Payload Envelope (SONET)
STM	Synchronous Transport Module
STS-1	Synchronous Transport Signal level 1
STS-3	Synchronous Transport Signal level 3
SUNI	Saturn User Network Interface
TCM	Transmit Clock Macro
TCS	Transmit Clock Synthesis
TBD	To Be Determined
TTL	Transistor Transistor Logic
TU	Tributary Unit
TUG	Tributary Unit Group
UNI	User Node Interface
UTOPIA	Universal Test & Operations PHY Interface for ATM
VCO	Voltage Controlled Oscillator
VC-4	Virtual container-4
WDT	Watchdog Timer

Table 4.1: Abbreviations

5 Introduction

For readers unfamiliar with SMART, a short introduction is given in the following.

SMART (Scalable and Modular ARchitecture for SONET/SDH Technology) represents a scalable SONET/SDH[3] processor macro that performs the Transmission Convergence sublayer functions and part of the Physical Medium Dependent sublayer functions (Figure 5.1). It has been developed at the IBM Zurich Research Laboratory.

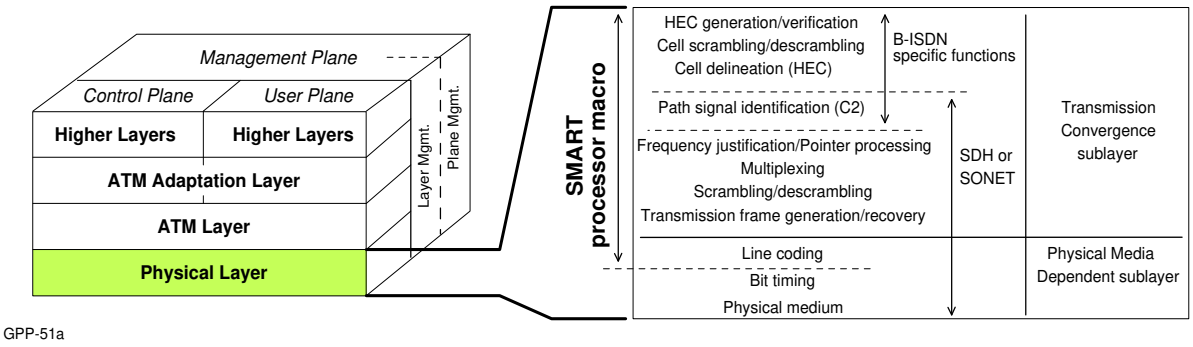

Figure 5.1: Physical Layer coverage

The processor macro processes the overhead, administrative unit pointers and payload for STS-1, STS3c, and STM-1 signals in both the receive and transmit direction. It maps ATM cell streams into the payload and vice versa. The mapping of other data is supported by disabling the ATM cell mapping and providing direct access to the virtual container (referred to as circuit switched mode, CS mode). Thus, a processor macro can be seen as a SONET/SDH processor for STS-1, STS-3c, and STM-1 signals and is ideally suited to applications in ATM terminals, switching systems, hubs, telecommunication network nodes, add/drop multiplexers (ADM) and digital cross-connect systems (DXC).

SONET/SDH processors for higher order signals (e.g. STS-3, STS-3c, STS-12, STS-12c, STS-24, STS-48, STM-4, STM-4c, STM-16, STM-64) are obtained by operating multiple processor macros in parallel. No additional stages with SONET/SDH related functions are needed. In general, the scalability to higher order signals is achieved by applying parallelism instead of running the macros at higher speed. Thus, modular and scalable designs with linear cost growth with respect to the standardized SONET/SDH signal hierarchies are supported.

The integration of four processor macros into one SMART chip is depicted in . It can be configured to process up to four STS-1, STS-3c, or STM-1 signals as well as one STS-3[4], STM-4, STS-12c, or STM4c signal by operating its four internal macros either individually or in parallel.

STS-12 (12x 51.84 Mbit/s), STS-24 (24x 51.84 Mbit/s) and STS-48 (48x 51.84 Mbit/s) are achieved by operating three, six and twelve chips, respectively, in parallel. STM-16 (16x 155.52 Mbit/s, 4x 622.08 Mbit/s) and STM-64 (64x 155.52 Mbit/s, 16x 622.08 Mbit/s) are achieved by operating four and sixteen chips, respectively, in parallel.

A SMART chip contains a UTOPIA Level 2 compliant ATM cell interface with up to 50 Mword/s[5] capacity. For up to four times STS-1, STS-3c or STM-1 as well as one time STS-3 or STM-4, the corresponding number of 6.48 Mbyte/s or 19.44 Mbyte/s ATM cell streams[6] are time division multiplexed across this

[3] In the following, referring to SDH refers in any case to SONET too.

[4] Three out of four macros operate in parallel.

[5] Word: 2 byte (16 bit). 50 Mword/s ≡ to 800 Mbit/s

[6] The SONET/SDH frame payload carries up to 49.536 Mbit/s (STS-1), 145.152 Mbit/s (STS-3), 149,76 Mbit/s (STS-3c, STM-1), and 599,04 Mbit/s (STM-4/-4c) user data (ATM cell mapping). For a quick cross-reference, 51 Mbit/s (6.84 Mbyte/s), 155 Mbit/s

interface (UTOPIA Level 2, Multi-PHY configuration, multiple PHY ports per PHY device). STM-4c can be operated either as Single- or Multi-PHY configuration. The ATM cell interface can be used as described above (UTOPIA Level 2) or can be configured optionally to operate as two independent UTOPIA Level 1 interfaces. The latter one enables up to two independent 19.44 Mbyte/s (6.84 Mbyte/s) ATM cell streams which can be processed by (cheap) 19.44 Mbyte/s (6.84 Mbyte/s) ATM layer chip-sets. Non-ATM mapping and direct access to a virtual container are provided by an additional, independent interface (CS interface) with up to 50 Mword/s[7] capacity for both the transmit and the receive direction[8].

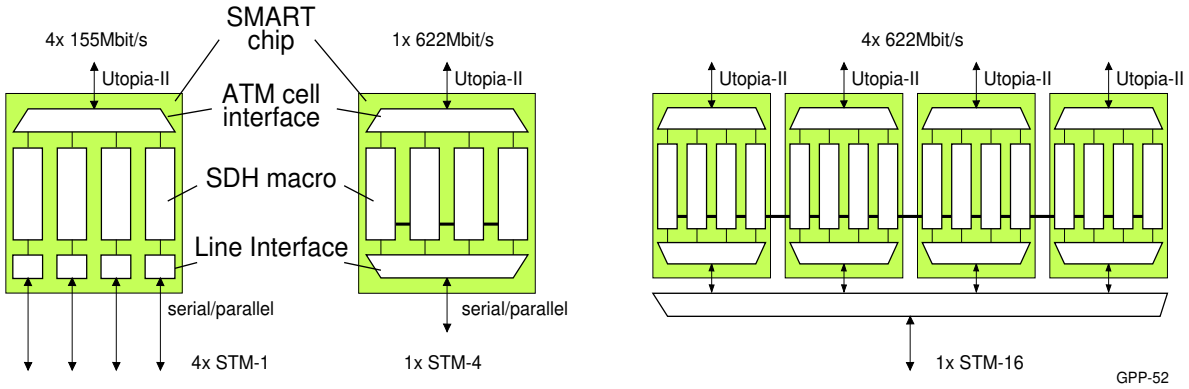

Figure 5.2: SMART application examples (generic)

As an example, depicts the internal structure of the four processor macros of the chip for the STM-4 transmit direction. The ATM Cell Interface (ACI) adapts between the UTOPIA Level 2 specification [1,2] and the on-chip ATM Cell Buffer (ACB) of the ATM Cell Handler (ACH). The Idle Cell Insertion (ICI) provides a continuous cell stream by generating idle/unassigned cells in case of an empty ATM Cell Buffer. The ATM Cell Scrambler (ACS) scrambles the payload of the cells (self synchronized scrambler, $1 + x^{43}$, [8]) and writes the cells into the SDH Buffer (SBU) of the Overhead and Frame Processor (OFP). The SDH Framer (SFR) provides the appropriate frame structure and maps the ATM cells into the frame payload [3, 4, 5, 6]. The SDH Frame Scrambler (SFS, frame synchronized scrambler, $1 + x^6 + x^7$, [8]) scrambles the SDH frame and forwards the SDH signal to the Line Interface (LI). The Line Interface provides bit and byte clocks derived by clock synthesis from a reference clock and multiplexes the 4x 8bit SDH signal (4x 19.44 Mbyte/s) to either a serial or 8bit parallel SDH signal (622.08 Mbit/s, 77.76 Mbyte/s).

(19.44 Mbyte/s), and 622 Mbit/s (77.76 Mbyte/s) are used for ATM cell streams carried by STS-1, STS-3/3c, STM-1, and STM-4/-4c signals, respectively.

[7] Word: 2 byte (16 bit). 50 Mword/s ≡ 800 Mbit/s

[8] Not included in Figure 5.3.

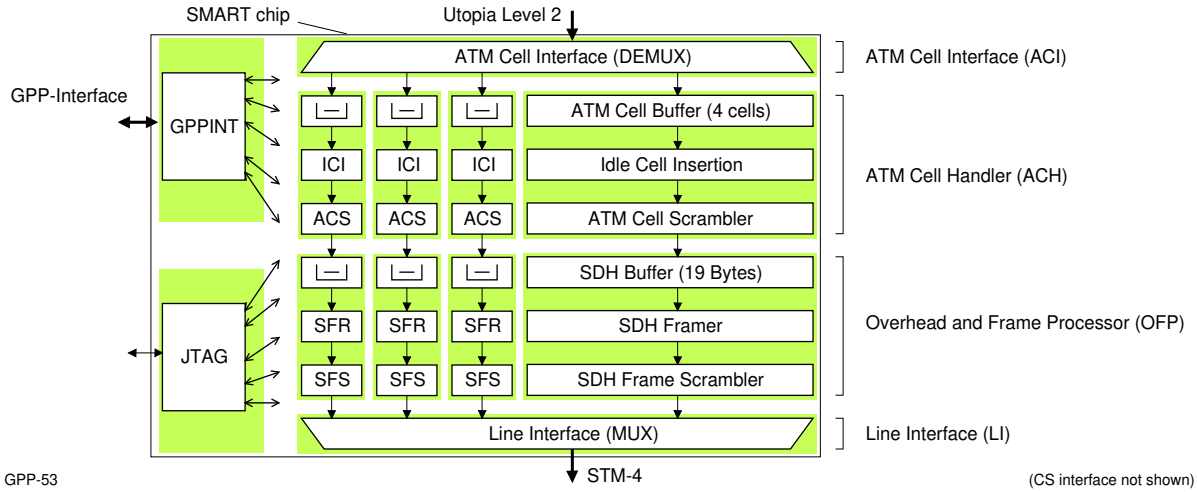

Figure 5.3: STM-4 application example (generic)

Via the General Purpose Processor Interface module (GPPINT) configuration data is written into the chip as well as status information and statistics are read out (Figure 5.4). The interfaces of this module and the organization of configuration data as well as status and statistic information are the target of this document.

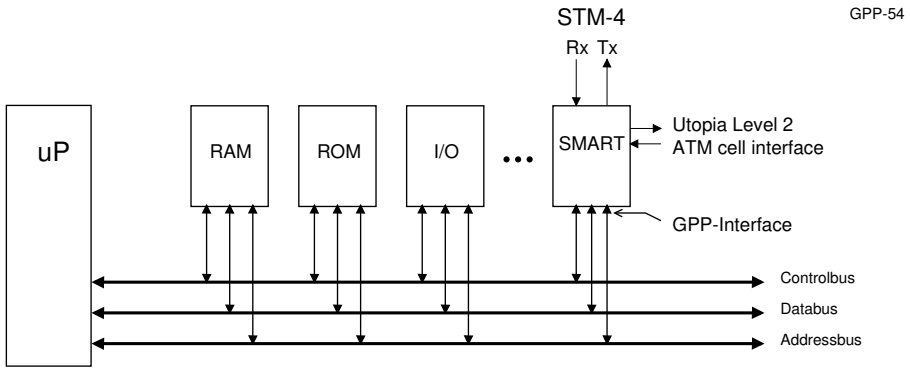

Figure 5.4: Embedding of SMART into a microprocessor environment (generic)

6 Overview

6.1 GPPINT, Chiplets and Clock Islands

Via the General Purpose Processor Interface module (GPPINT) configuration data is written into the chip as well as status information and statistics are read out. The GPPINT chiplet is connected to all other functional modules (chiplets) like the Line Interface, ATM Cell Interface, ATM Cell Handler or Overhead and Frame Processor, see . It has two types of interfaces: one chip-external and several chip-internal interfaces. The chip-internal interfaces connect to all other functional modules of the chip and take into account the various asynchronous clock islands of the chip. A clock island represents a logic or silicon area with one designated clock. As described in more detail below it consists of one chiplet, a group of chiplets or parts (components) of chiplets with the same clock; thereby the grouping of the chiplets is depending on the selected configuration. The chip-external interface is described in chapter 9.

Configuration registers provide control signals that are used in the functional modules to control e.g. Finite State Machines (FSM), counters, data manipulation, data flow and Growable Register Arrays (GRA, RAM areas). There are two types of configuration data: static and non-static signals. Static configuration data is typically set once after reset. During normal operation it does not change and has no dependency on any clock. Non-static configuration data is set after reset but may be changed during normal operation to affect the current operation hitless on the fly. Thus, it has a dependency on the clock of the considered chiplet. For non-static configuration data, a buffered transfer into the chiplet must be provided to avoid metastability problems.

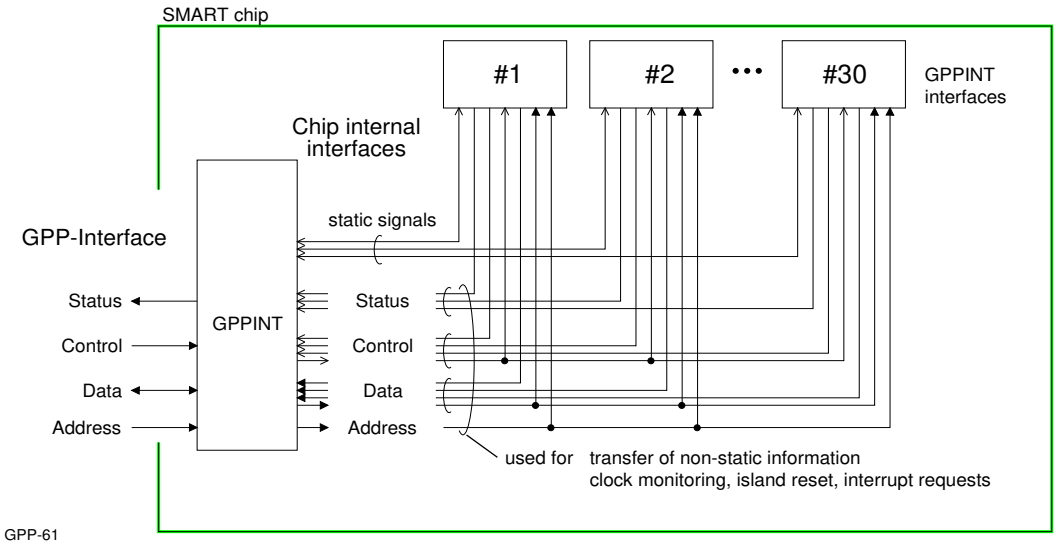

Figure 6.1: General Purpose Processor Interface module (GPPINT)

The GPPINT module of the SMART chip writes the non-static configuration data of a chiplet into unique configuration registers inside the considered chiplet. A bi-directional handshake mechanism between the GPPINT and each chiplet controls the data transfer without metastability. All static configuration signals are provided by the GPPINT module and are distributed to all chiplets where needed (Figure 6.1).

Status information (status signals) represents previously written configuration data, the current status of e.g. Finite State Machines, counters, data manipulation, data flow, SDH links and GRAs. Status signals typically change during normal operation and have a dependency on the clock of the considered chiplet. Thus, they must be treated as non-static signals and a transfer of status information out of a chiplet must be buffered to avoid metastability. The GPPINT module reads the status information out of registers,

counters and GRA areas inside each chiplet. The same bi-directional handshake mechanism (as used for writing configuration data) controls the data transfer without metastability.

This document specifies the transfer of non-static configuration data and status information between GPPINT and the other chiplets. Static GPPINT configuration and status signals as well as non-static GPPINT signals are listed in a separate document.

As an example, Figure 6.2 and Figure 6.3 show the functional modules and the clock islands for 4x STM-1 and 1x STM-4/-4c respectively. Note, the ATM cell or SDH data transfer from one clock island to another is done via buffer stages. From a logical point of view, the buffer stages are associated to a specific functional module. The ATM Cell Buffer and the SDH Buffer are associated to the ATM Cell Handler and the Overhead and Frame Processor respectively. But writing into and reading out of one of the buffers is done from different clock islands. Note, an 'Overhead and Frame Processor' module is always locked to its associated 'Line Interface' module (always same clock source). Thus, throughout this document, referring to the 'Overhead and Frame Processor' module/clock island refers always to the associated 'Line Interface' module too.

Figure 6.2 represents an example with ten asynchronous clock islands. This is because of four Overhead and Frame Processor receive clock islands (individually clocked), each associated to a logical OFP receive module, and the configuration of SMART to one Utopia Level 2 ATM cell interface.

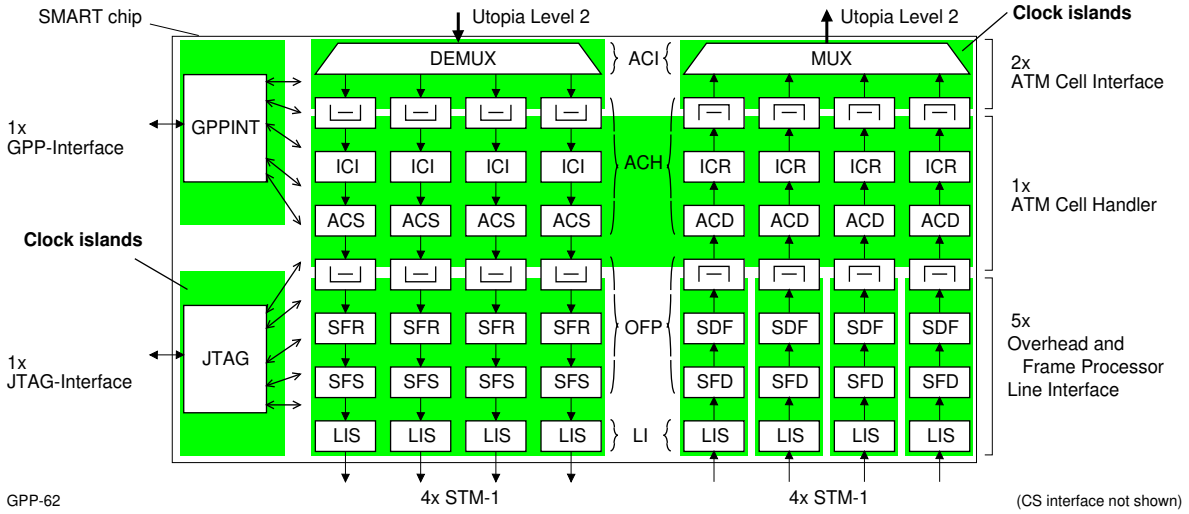

Figure 6.2: 4x STM-1: Functional modules (chiplets) and clock islands

Figure 6.3 is different (seven clock islands) due to one Overhead and Frame Processor receive clock island covering all four logical OFP receive modules. This does not result in an additional GPPINT interface. Although clocked by one common clock, the GPPINT module accesses the four logical Overhead and Frame Processor receive chiplets (which operate in parallel) via their individual interfaces.

The same interface structure is defined for the Overhead and Frame Processor transmit direction and the ATM cell Handler (receive and transmit direction). This solution allows the continuation of the modular design approach. Between the ATM Cell Interfaces and the ATM Cell Buffers four Port Handlers (PH) are realized for the receive and for the transmit direction. Depending on the configuration of the chip, these port handlers are configured to one out of five different clock sources, thus individual GPPINT interfaces are provided for each of the handlers. This is summarized in Figure 6.4 by showing all thirty GPPINT

interfaces. [Presently, no handshaking interfaces are provided for the CS_Tx and CS_Rx chiplets --> only 28 interfaces]

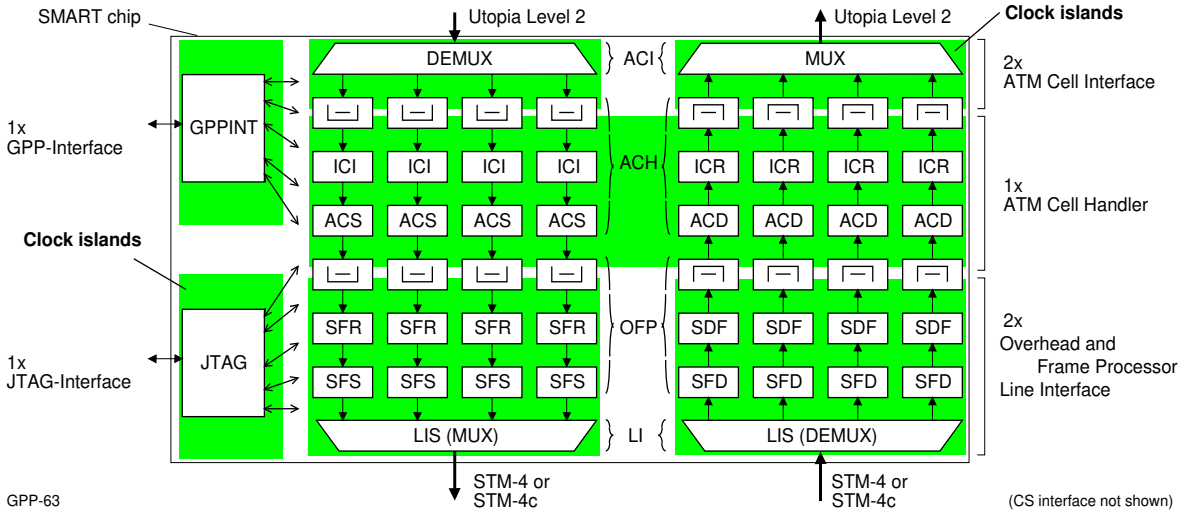

Figure 6.3: 1x STM-4/-4c: Functional modules (chiplets) and clock islands

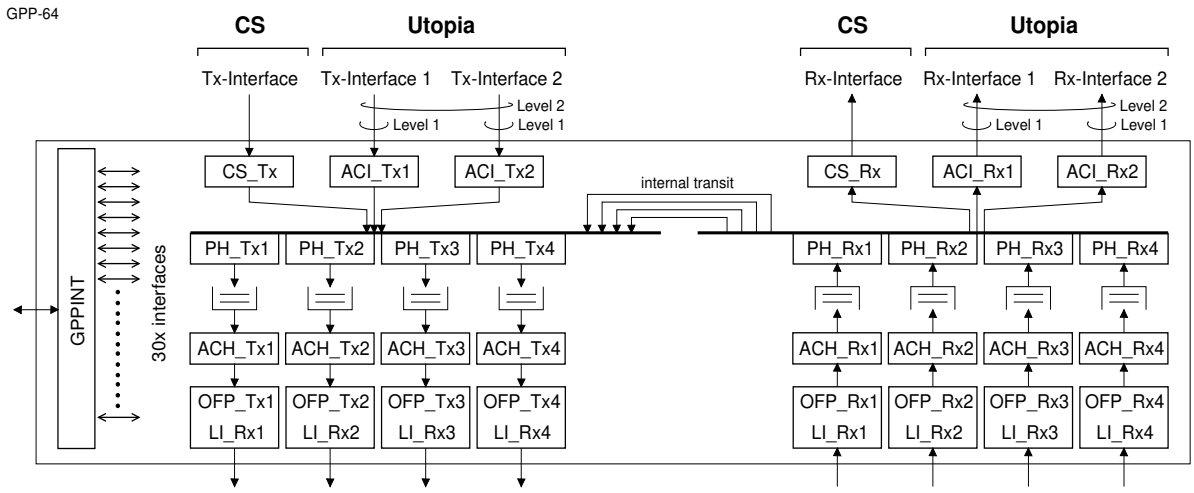

Figure 6.4: GPPINT chip-internal interfaces

Configured to either one Utopia Level 2 or two Utopia Level 1 ATM cell interfaces, Table 6.1 lists and summarizes the number of clock islands for the receive and transmit directions for some configuration modes of the SMART chip. Table 6.2 lists the corresponding numbers of used GPPINT interfaces. Both tables cover pure ATM, pure CS, mixed ATM/CS, ADM, DXC and mixed ADM/DXC configurations of the SMART chip.

	Configuration modes															
	1x Utopia Level 2 interface										2x Utopia Level 1 interfaces					
Number of Clock Islands	**K x STS-1**		**K x STS-3c**		**K x STM-1**		**1x STM-4**		**1x STM-4c**		**K x STS-1**		**Kx STM-1**		**1x STM-4**	
Chiplets	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**
CS	1	1	1	1	1	1	1	1	0/1	0/1	1	1	1	1	1	1
ACI	1	1	1	1	1	1	1	1	1/0	1/0	2	2	2	2	2	2
ACH	1		1		1		1		1		1		1		1	
OFP/LI	1	K	1	K	1	K	1	1	1	1	1	K	1	K	1	1
GPPINT	1		1		1		1		1		1		1		1	
JTAG	1		1		1		1		1		1		1		1	
Clock islands/chip	8 + K		8 + K		8 + K		9		7		10 + K		10 + K		11	

Table 6.1: Number of clock islands (examples)

	Configuration modes															
	1x Utopia Level 2 interface										2x Utopia Level 1 interfaces					
Number of used GPPINT Interfaces	**K x STS-1**		**K x STS-3c**		**K x STM-1**		**1x STM-4**		**1x STM-4c**		**K x STS-1**		**Kx STM-1**		**1x STM-4**	
Chiplets	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**	**Tx**	**Rx**
ACI	1	1	1	1	1	1	1	0	0	1	2	0	2	2	2	2
PH	4	4	4	4	4	4	4	4	4	4	4	4	4	4	2	2
ACH	K	K	K	K	K	K	4	4	4	4	K	K	K	K	2	2
OFP	K	K	K	K	K	K	4	4	4	4	K	K	K	K	2	2
Internal interfaces	10 + 4K		10 + 4K		10 + 4K		25		25		10 + 4K		12 + 4K		16	

Table 6.2: Number of used GPPINT interfaces (examples)

Although listed in Table 6.1 (for completeness reasons), the GPPINT does not access the JTAG module[9] and the CS modules. To support the configuration of e.g. 4x STM-1, (K = 4) together with two Utopia Level 1 ATM cell interfaces, the GPPINT module provides up to 28 chip internal interfaces.

Note, both tables show different examples and do not cover all possible configurations.

[9] There is no data interconnection between the GPPINT and JTAG module.

6.2 APS: Automatic Protection Switching

Automatic Protection Switching (APS), as defined by ITU recommendations G.783, see [5], and Bellcore document SR_NWT-001756, see [7], is supported by means of two 5x5 cross-connects (referred to as APS cross-connect) and two bi-directional APS chip interfaces. For the transmit and receive direction, an APS cross-connect is provided between the SDH Buffer and the ATM Cell Handler respectively, see Figure 6.5. With this partitioning, all ports of the cross-connects are located inside one clock island. This allows protection switching on the fly without the need of switching clocks (metastability !). Furthermore, protection switching can be performed regardless of ATM cell and internal/external CS modes of operation or any combination. Note, the APS cross-connect for the transmit direction supports APS bridging and switching.

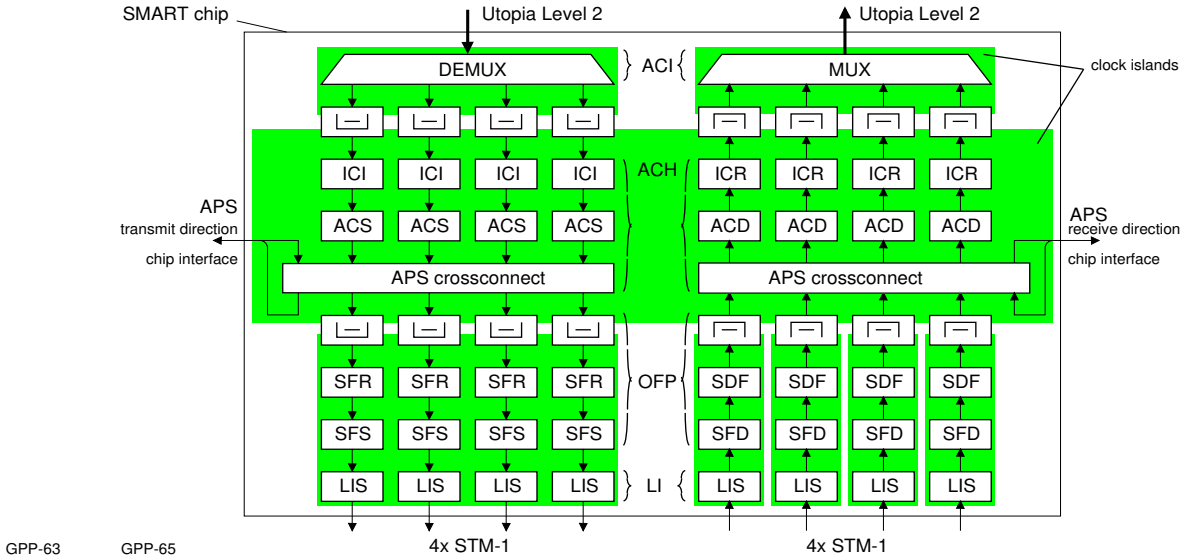

Figure 6.5: APS cross-connect stages and chip-external interfaces

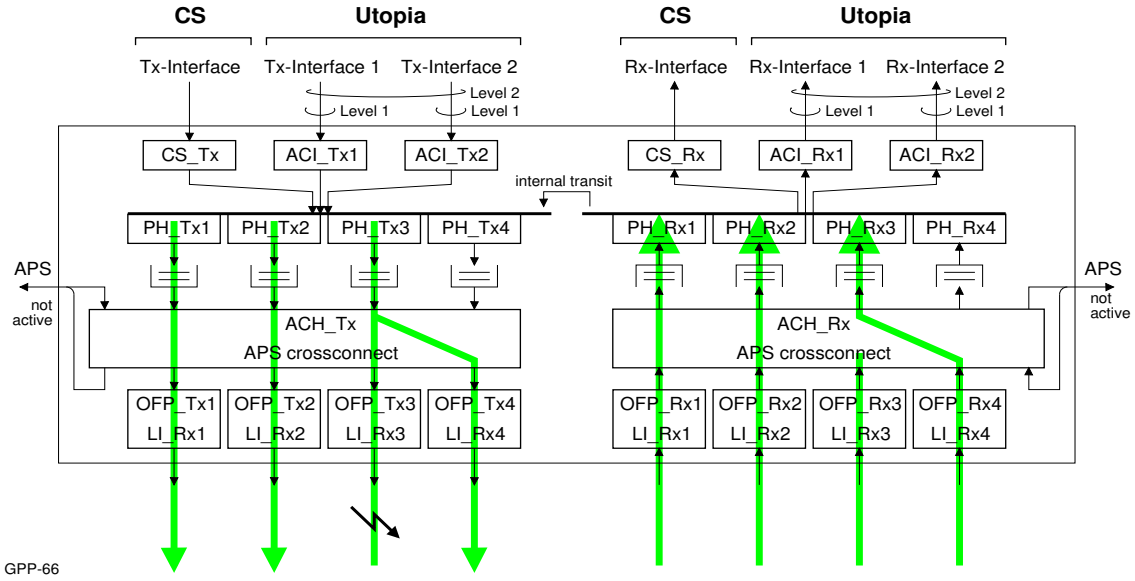

Figure 6.6: 1:4 protection with one SMART chip

An APS cross-connect has five input and five output ports (5x5 structure). Four ports are used to switch between the four SDH macros on a SMART chip, see Figure 6.6 (1:4 protection). Operating multiple SMART chips in parallel, an 1:N protection is achieved activating the fifth APS cross-connect port and the APS external interfaces, see Figure 6.7. The chips are all connected to two bi-directional 8bit busses. Each bus is driven, at a time, only by one chip in a semi-permanent mode of operation[10]. With four SMART chips, this interconnection principle can be applied to up to 14 working lines and one protection line.

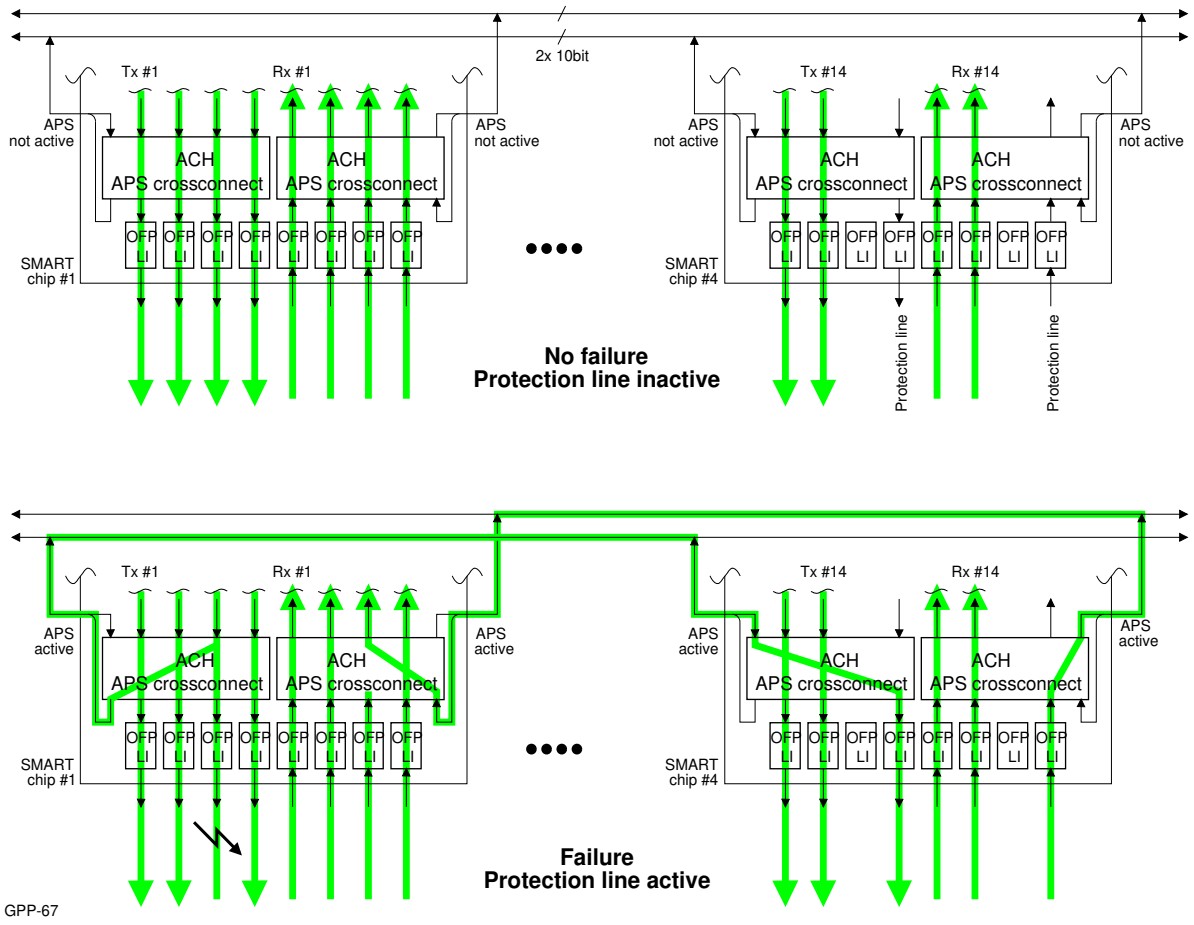

Figure 6.7: 1:N protection with multiple SMART chips

10 There is no high speed switching of the bus direction or between different bus drivers (chips) required. A bus is continuously driven by one chip as long as the protection line is used.

7 GPPINT Chip Internal Interfaces

7.1 Functional Description and Timing

Figure 7.1 shows the principal interconnections between the GPPINT and the other chiplets of the SMART chip used for the transfer of non-static information[11]. The GPPINT module outputs an address bus (10bit, GPPAddr[0:9]), a data bus (8bit, GPPData[0:7]) and a control signal (read/write, GPPRnW) common to all SMART chiplets. Configuration data is written via this data bus into the chiplets (write cycle). Status information and counters are read out via an individual data bus (read cycle) for each chiplet. All read and write transfers are controlled by two control signals for each chiplet: DS (Data Strobe, GPPINT to chiplet) and DTACK (Data Transfer Acknowledge, chiplet to GPPINT).

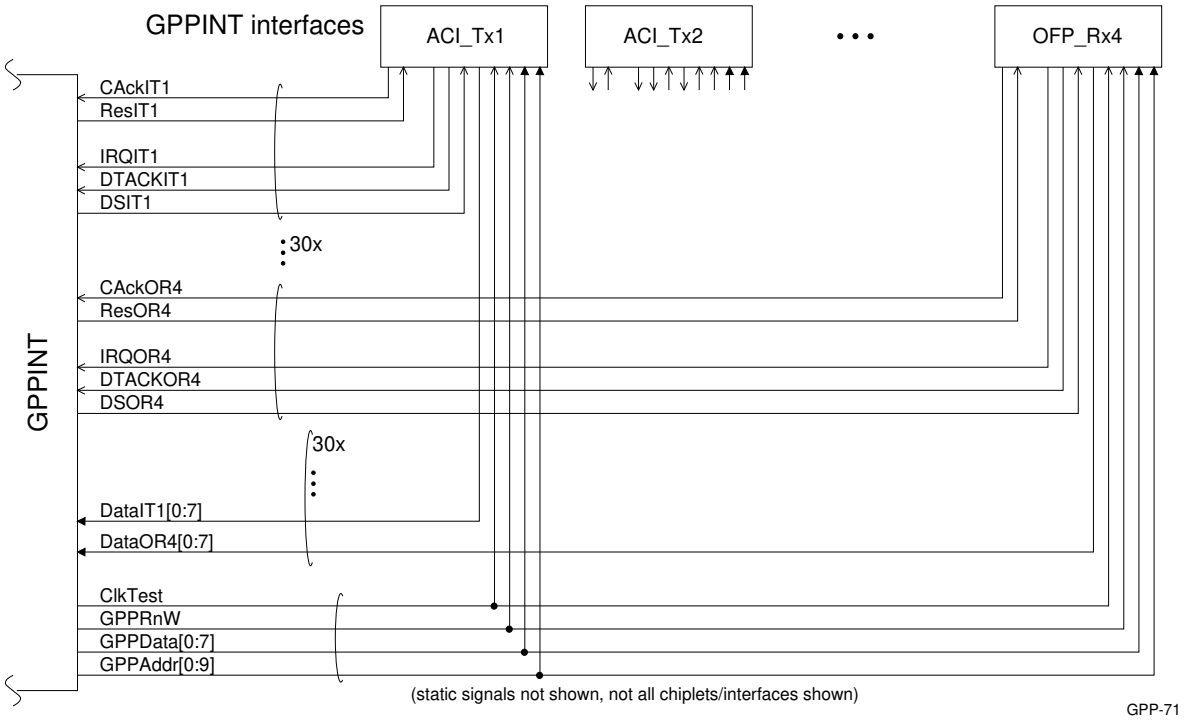

Figure 7.1: GPPINT chip-internal interface signals

Each chiplet outputs an additional status signal to the GPPINT: Interrupt Request (IRQ). IRQ reports and requests an interrupt to the external microprocessor. More details can be read out from a dedicated IRQ status register inside the requesting chiplet. Each defined bit position represents a specific cause. The reset conditions for these signals are individual for every request. More details can be found in the description of the considered chiplet. The GPPINT module may mask the interrupt request signals of the chiplets by a GPPINT internal interrupt mask register. In addition, a chiplet may mask its internal interrupt events by a chiplet internal interrupt mask register. A detailed description of the interrupt architecture of SMART is given in chapter 7.4.

The GPPINT module buffers all incoming DTACK and IRQ signals by a 2-stage pipeline (2-stage buffer), see . A chiplet buffers its incoming DS control signal by a 2-stage pipeline too. Furthermore, GPPINT synchronizes its outgoing GPPAddr[0:9]), GPPData[0:7]) and GPPRnW signals to its clock CClkGPP by at least one register stage. A chiplet synchronizes its outgoing data bus to its chiplet clock by at least one register stage.

[11] See chapter for additional, individual and buffered chiplet signals (e.g. clock monitoring).

For simplicity reasons, Figure 7.2 depicts the output registers for the control and status signals GPPRnW, DS, DTACK and IRQ as individual, additional registers. Typically, these signals are derived from the FSMs states and may be driven e.g. directly by a FSM state register.

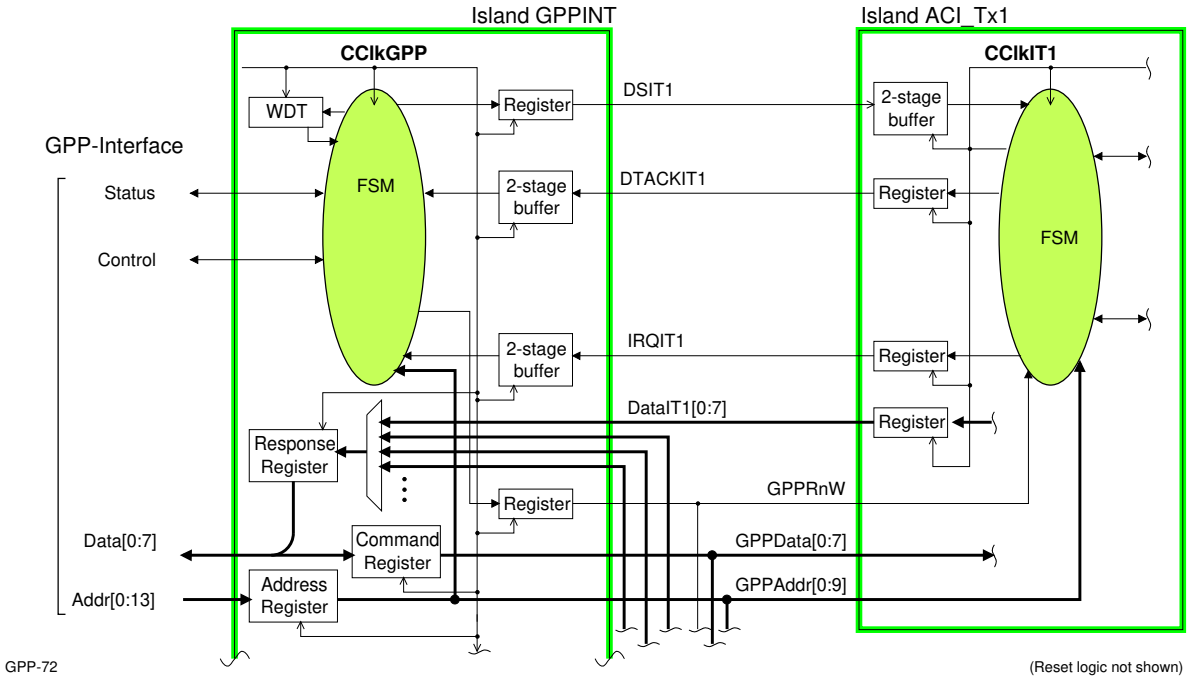

Figure 7.2: Details of signal buffering (chiplet ACI_Tx1 as example)

The GPPINT module requests a data transfer by asserting the corresponding DS signal. A chiplet acknowledges by asserting its DTACK signal. For a write cycle, it asserts DTACK not before it has clocked the data into its internal register. For a read cycle, it asserts DTACK not before it has output the data on its data bus. Examples of timing diagrams for a single read and write cycle are given in Figure 7.3 and Figure 7.4 respectively. Note, a transfer to/from a chiplet can only be requested (started) by the GPPINT if the corresponding DTACK signal is deasserted.

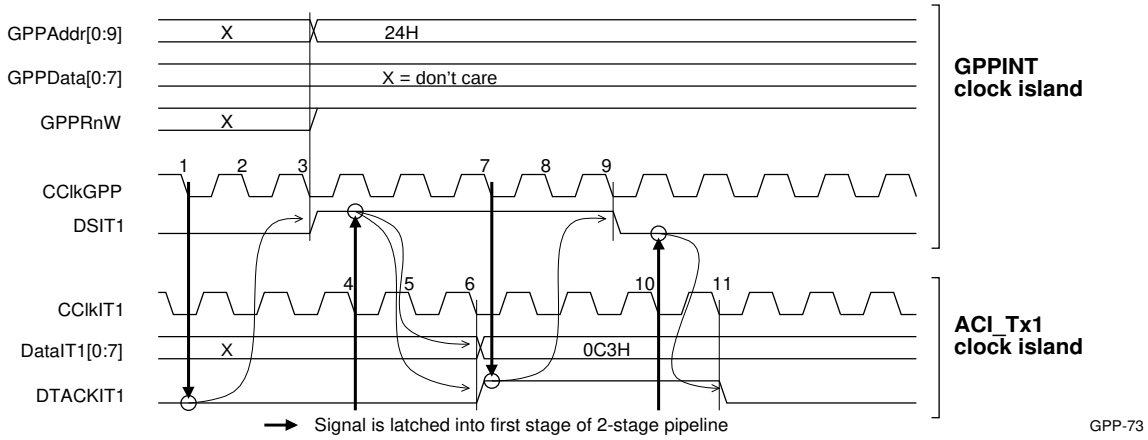

Figure 7.3: Read cycle timing (example)

Common for read and write cycles, the GPPINT only changes GPPAddr[0:9] and GPPRnW together with the rising edge of a DS signal, otherwise the address bus and the control signal stay stable. For a write cycle, GPPData[0:7] changes only together with the rising edge of a DS signal, otherwise it stays stable. For a read cycle, a chiplet only changes its outgoing data bus together with the rising edge of its DTACK signal.

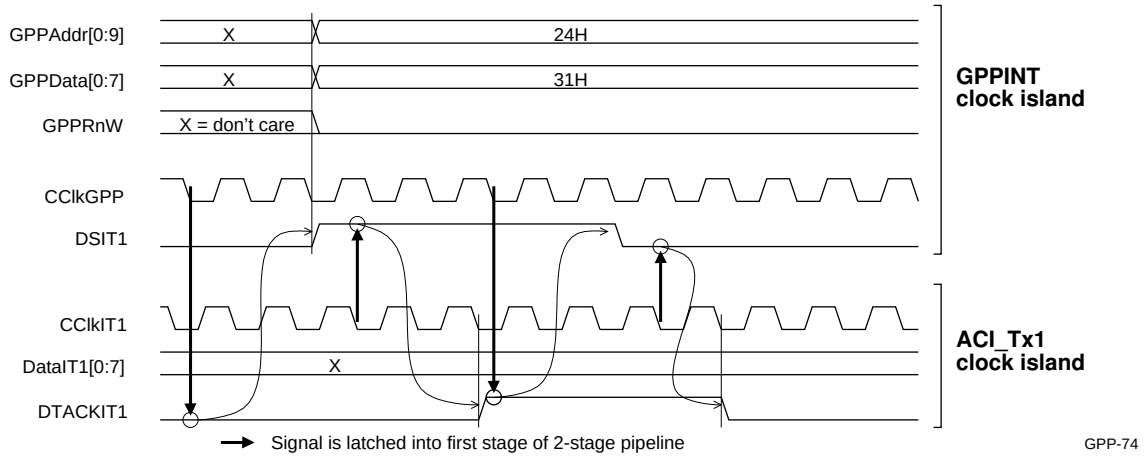

Figure 7.4: Write cycle timing (example)

The GPPINT module provides a full buffering of the external microprocessor address, data and control bus. This allows a separation of the activities on the GPPINT chip-internal and external interfaces (clock fire-wall). It allows the termination of the processor write cycle without the need for waiting until the internal chiplet has also terminated the cycle and prevents the external microprocessor of being completely stopped by hardware in case of addressing an un-clocked SMART chiplet. To further prevent a hang-up of the microprocessor, a watchdog timer (WDT) is started when the SMART chip is selected by the microprocessor. Upon time-out, the GPPINT terminates both, the current internal and the external cycles by itself and indicates this event by setting an appropriate interrupt request bit.

The time-out period of the watchdog timer can be configured by the microprocessor in multiples[12] of the GPPINT clock CClkGPP period. Upon external reset, it is set to its highest possible value and must be modified by the microprocessor if needed. If a time-out event occurs, the watchdog timer is automatically preset to its highest value to prevent a possible deadlock and must be reconfigured by the microprocessor if needed.

Note, buffering the microprocessor address and data busses by the GPPINT does not cause an additional delay. This is hidden by the GPPINT clock cycles required to synchronize the incoming GPPINT chip-external or -internal control signals.

In all the examples, fastest FSM reaction and an undisturbed read/write access to any selected register and GRA area inside a chiplet are assumed. The two-stage pipeline for the DS signal causes a minimum delay of two clock cycles until a chiplet FSM is able to react/respond to a request. For a read cycle, the synchronization register at the chiplet data bus output adds one additional chiplet clock cycle until the valid data is finally output to the GPPINT. For a write cycle, one additional chiplet clock cycle is needed to clock inside the chiplet the data into the selected configuration register or GRA. These dependencies are detailed and numbered in Figure 7.3 as follows:

1) The GPPINT latches DTACK "low" into its first pipeline stage

[12] Range: 1 to 255. Value 0 is treated as 255.

2) The GPPINT latches DTACK "low" into its second pipeline stage, now available for the GPPINT FSM. The FSM prepares a state transition to start a read cycle at the next clock edge of CClkGPP
3) The GPPINT outputs valid GPPAddr, GPPRnW "high" (read cycle) and DS "high" (asserted)
4) The chiplet latches DS "high" (asserted) into its first pipeline stage
5) The chiplet latches DS "high" (asserted) into its second pipeline stage, now available for the chiplet FSM. The FSM prepares a state transition and accesses the selected register/GRA location for data to be valid on the outgoing data bus at the next clock edge of the chiplet's clock
6) The chiplet outputs the valid data on the outgoing data bus and asserts DTACK ("high")
7) The GPPINT latches DTACK "high" (asserted) into its first pipeline stage
8) The GPPINT latches DTACK "high" (asserted) into its second pipeline stage, now available for the GPPINT FSM. The FSM prepares a state transition to terminate read cycle at the next clock edge of CClkGPP
9) The GPPINT outputs DS "low" (deasserted)
10) The chiplet latches DS "low" (deasserted) into its first pipeline stage
11) The chiplet latches DS "low" (deasserted) into its second pipeline stage, now available for the chiplet FSM. The FSM prepares a state transition to terminate the read cycle at the next clock edge of the chiplet's clock. Simultaneously, it outputs DTACK "low" (deasserted). This terminates the cycle.

Figure 7.5 shows a read cycle followed by a write cycle from and to the same chiplet. The marked area indicates a time interval which depends on the actual microprocessor timing and may be multiple GPPINT clock cycles. The figure further shows that a cycle at the chip internal interface must be completely terminated[13] before a next cycle to the same or any other island can be initiated by the GPPINT.

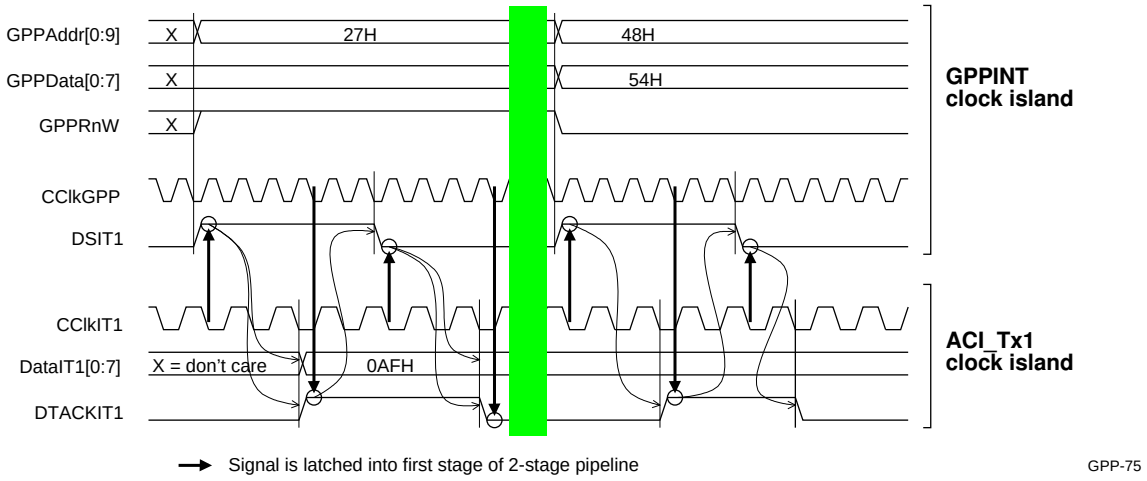

Figure 7.5: Back to back read and write cycles from and to the same chiplet (example)

A substantial number of wait cycles of the microprocessor may be needed in case the clock frequency of an accessed chiplet is low. For a read access the internal and external cycles complete simultanously. After latching address and data in a write access the external cycle is completed, thereas the internal cycle may continue for some time. If a next access follows a write cycle immediately, the microprocessor inserts wait state cycles until the internal cycle of the last access has been completed.

[13] All DS and all DTACK signals must be low.

7.2 Chiplet Reset

The reset operation of the SMART chip is performed in two stages, see Figure 7.6. An external hardware signal (chip RESET, active low) resets the GPPINT module and seven reset registers (56 bit). The bits of these registers are used as reset signals for the individual chiplets (e.g. ResetIT1, active high). Each chiplet uses this signal as reset signal for its own reset register(s). Some chiplets need more than one reset signal for controlled start-up of internal functional sections. The chiplet internal logic is reset upon the chiplet reset registers. The chiplets that have an internal interface (GPPHandler) have a 2 bit reset register (read/write). Bit 7, the chiplet Reset control signal, is active high after power-on-reset, causing the chiplet to be disabled. Bit 6 is the chiplet Halt signal, which is used to freeze the state machines (e.g. Handshake_FSM, Reset_FSM1, Reset_FSM6) of the chiplet for diagnostic purposes. This allows a complete hardware controlled chip reset by an external signal and further selective, microprocessor controlled reset, enable and disable of dedicated functions inside the chiplets.

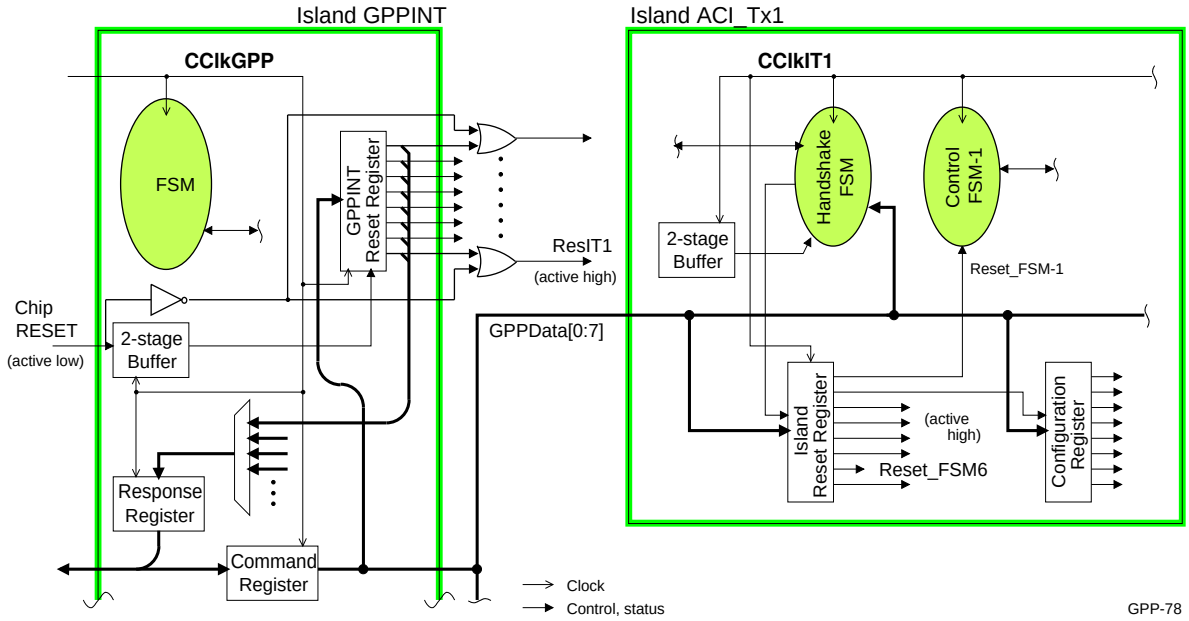

Figure 7.6: SMART reset principles (example, one chiplet only)

The starting sequence of the SMART chip from reset is as follows:

1. The active external chip reset signal forces a reset of all reset registers in all chiplets[14].
2. Before the external chip reset can be released at least the GPPINT clock must be applied.
3. After deassertion of the external reset (low-to-high transition), the microprocessor configures all static signals in the GPPINT registers (signals to all other chiplets). In particular, the configuration bits that control the clock multiplexing must be written at this stage.
4. The microprocessor selectively enables individual chiplets by deactivating (setting low) the corresponding outgoing reset signals (active high) to the individual chiplets (writing the appropriate pattern into the GPPINT reset registers). This requires that the clock of the particular chiplet is applied before.
5. The microprocessor configures the non-static configuration signals inside the enabled chiplet.
6. The microprocessor selectively enables individual functions inside each enabled chiplet by deactivating appropriate bits in the reset register (writing the appropriate pattern into the chiplet reset registers) or by setting specific configuration bits (enable/disable) inside the chiplet.

[14] Even if NO clock is running.

The GPPINT module can monitor a particular chiplet's clock before deactivating the chiplet reset signal(s) to make sure that the chiplet clock is active. Clock monitoring is described in the next chapter. Except the Line Interface Module[15], the GPPINT reset signals are buffered inside the chiplets. Thus, a chiplet may not reset properly if the chiplet reset signal is deasserted by the GPPINT while the chiplet clock is not yet available (running). Furthermore, the Handshake FSM receives directly the GPPINT reset signal and is not controlled by a bit of the reset register.

The reset registers behave like other configuration registers. Note, an active chiplet reset signal supersedes all setup of other configuration registers.

[15] This allows a reset of the LIM (SIM and PIM chiplets) and the setup of the clocks at all, see [11].

7.3 Clock Monitoring

The GPPINT provides monitoring of all other chiplet clocks, see Figure 7.7. The GPPINT module periodically toggles a test signal (ClkTest) delivered to each clock island. After one period of ClkTest (CMP, clock monitor period), which has to be configured again for each chiplet, it tests the returning acknowledge signal and compares it to the test signal. If a difference is detected the GPPINT indicates a loss of island clock by setting the corresponding status bit in the clock status register. The stored status bits can be masked by a clock monitor status mask register. With at least one unmasked, active status bit, an interrupt request is output to the interrupt mask function of the GPPINT.

As there is only one global clock test period for testing all clock islands it may be necessary to reconfigure the test period for a specific chiplet or a group of chiplets. The test period is derived from the CClkGPP by a configurable divider providing a ratio range of 1:1 to 1:255[16]. With a 50 MHz CClkGPP frequency[17] this results in maximal 5.1æs for the clock test period, covering island clocks ò196.078kHz.

Note, the Utopia interface specification allows a frequency range from DC up to 50MHz. In general, islands with a clock below 197KHz should always be masked by the Monitor Status Mask Register to avoid incorrect interrupt generation.

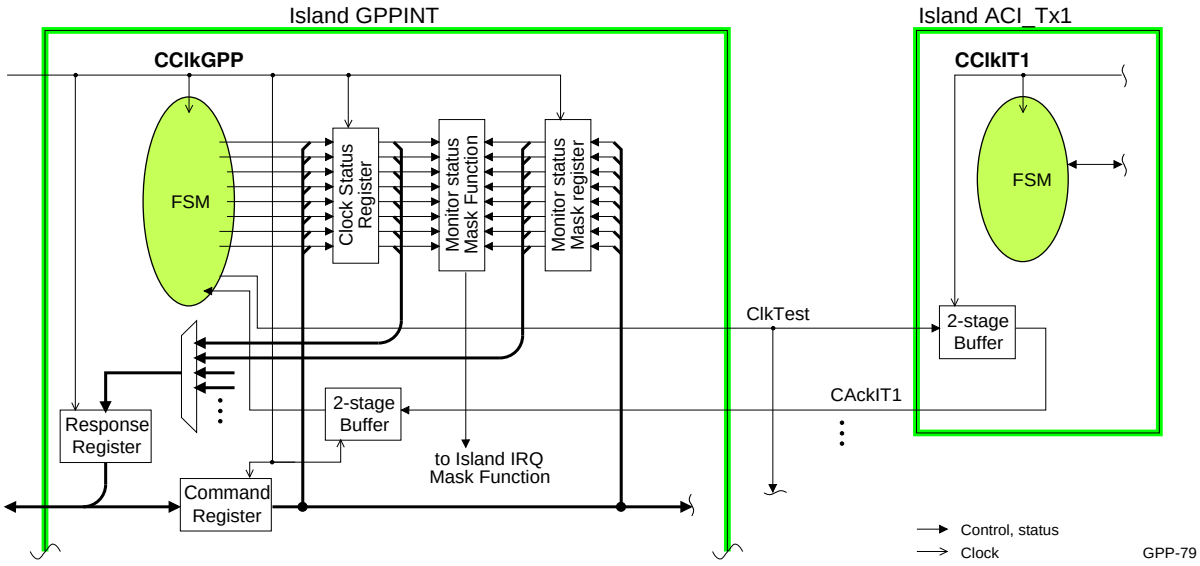

Figure 7.7: Chiplet clock monitoring principles (example, one chiplet only)

As mentioned above, the GPPINT provides monitoring of all other chiplet clocks. In case of multiple interfaces to one clock island like OFP_Tx and ACH_Tx/ACH_Rx, only one interface provides the monitoring function, see Table 7.1.

	OFP transmit clock island				**ACH clock island**			
GPPINT interfaces to chiplets	**OFP_Tx1**	**OFP_Tx2**	**OFP_Tx3**	**OFP_Tx4**	**ACH_Rx1 ACH_Tx1**	**ACH_Rx2 ACH_Tx2**	**ACH_Rx3 ACH_Tx3**	**ACH_Rx 4 ACH_Tx4**
Clock monitoring function	yes	no	no	no	no yes	no no	no no	no no

Table 7.1: One clock monitoring function per clock island

[16] A divider configuration value of 00H completely disables the clock monitoring.

[17] The CClkGPP frequency is specified from 10MHz to 50MHz.

7.4 Interrupts

The interrupt processing is performed in several stages, see Figure 7.8. An interrupt request inside a chiplet is stored into the interrupt pending register of the chiplet. The stored requests can be masked by an interrupt mask register. With at least one unmasked pending interrupt request in an 8-bit register a pointer bit in the chiplet's MAIN interrupt register is activated. The bits of the MAIN interrupt register can be masked too. If there is at least one active, unmasked pointer bit in the MAIN interrupt register, the chiplet outputs a chiplet interrupt request to the GPPINT module.

Bit 7 of the chiplet's MAIN interrupt register is always reserved and only activated when a fatal interrupt event occurs in the particular chiplet. See the individual chiplet descriptions for definitions of fatal events.

Writing into a chiplet's interrupt mask register, the microprocessor enables/disables individual interrupt requests. The microprocessor can read and write all interrupt pending registers and interrupt mask registers in all chiplets, except in the GPPINT (see below). After power-on-reset, all interrupts are masked.

Two mechanisms are provided to reset pending interrupt requests in a chiplet:

- ◊ Performing a write operation into the interrupt pending register with a specific reset pattern. The control logic however does not simply overwrite the register contents with this reset pattern. For each bit positions set in the reset pattern, the corresponding bit positions in the interrupt pending register is reset (set to zero). This avoids unintended loss of pending requests. The reset procedure can be performed for the requesting chiplet only. In the GPPINT module, the pending chiplet requests follow the incoming request signals after two GPPINT clock cycles, see Figure 7.8.
- ◊ Reading the pending interrupt request register automatically resets the register (all bit set to zero).

A reset can be always performed by the write operation. The optional automatic reset upon reading the register can be enabled/disabled by a non-static configuration signal.

In the GPPINT module, the chiplet requests can be masked by setting the chiplet interrupt mask register. With at least one unmasked pending chiplet interrupt request in an 8-bit register, a pointer bit in the MAIN interrupt register of the GPPINT is activated. The bits of the MAIN GPPINT interrupt register can be masked too. If there is at least one active, unmasked pointer bit in the GPPINT's MAIN interrupt register, the GPPINT outputs an interrupt request to the microprocessor (active low, tri-state otherwise).

In contrast to the other chiplet's interrupt pending registers, the pending chiplet request registers in GPPINT are read-only. The bits of these registers as well as the bits of the MAIN interrupt register in GPPINT are reset automatically by resetting or masking the responsible bits in the chiplets.

In addition to the pending chiplet request registers the GPPINT has two sets of pending interrupt request and masking registers that are treated like those of an ordinary chiplet: the Handshaking Error indication registers and the Clock Status registers. There is a MAIN handshaking error register with pointers to the other pending handshaking error registers and a MAIN clock monitor register with pointers to the other clock status registers. An active, unmasked bit in the MAIN handshaking error register activates a pointer bit in the GPPINT's MAIN interrupt register, and so does an active, unmasked bit in the MAIN clock monitor register. The pending handshaking error and clock status registers can be reset by read/write operations as described above for the chiplet's pending interrupt registers.

With this multi-stage architecture time consuming polling of each interrupt request register can be avoided. The microprocessor determines the cause of an interrupt request by first reading the MAIN interrupt request register of the GPPINT. The pointer from this register determines which register to read next: either a pending chiplet interrupt request register or the MAIN handshaking or MAIN clock status interrupt register. The pointer in a pending chiplet interrupt request register determines which requesting

chiplet's MAIN interrupt register to read next (inside that chiplet). Finally, the microprocessor reads the pending interrupt request register pointed to by the chiplet's MAIN interrupt register.

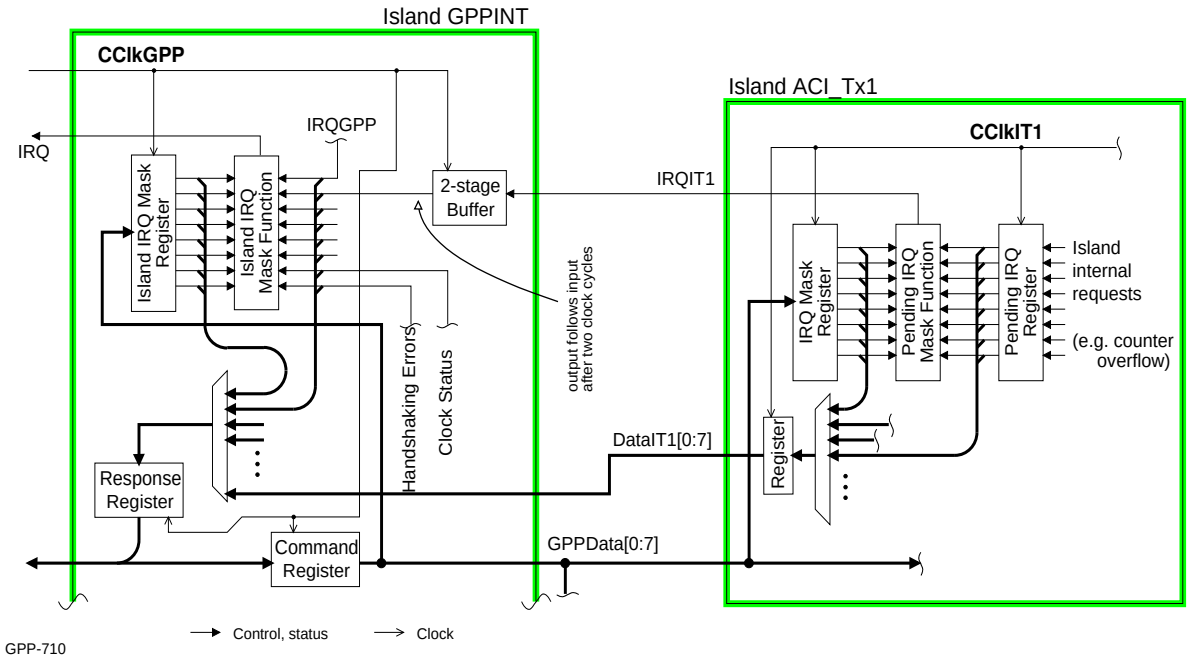

Figure 7.8: Interrupt processing principles (example, one chiplet only)

7.5 Statistic and Error Counters

Statistics and error conditions are gathered and counted inside the individual chiplets by counters. There are counters with widths between 8 and 24 bits. Counters with a width of >8bit are treated slightly different compared to 8bit counters.

Counters with 8bits (referred to as short counters) can be completely read by the microprocessor via the GPPINT by one read access (read cycle). Counters with >8bits (referred to as long counters) must be read in 8bit quantities (bytes) with multiple read cycles. Thus, during two successive read cycles a counter-internal overflow between two bytes may occur, and provisions against the reading of false counter values have to be made. When the microprocessor reads the lowest significant byte of a long counter the higher order bytes are stored in one or two read-on-the-fly registers, which are common for all long counters of a particular chiplet. In a second step, the contents of the read-on-the-fly registers is output to the GPPINT in individual read cycles.

The following operations and options are available for all counters of a chiplet:

- ◊ Upon chiplet reset, all counters are reset to zero. All threshold registers are set to their default value.
- ◊ All counters are read-only.
- ◊ Independent of the counter width, given that a counter has chiplet address offset **n** as a base, reading address **n** or address **n+1** both yield the **least** significant byte of the counter. Reading address **n** has no effect on the counter but reading address **n+1** will reset the counter after read.
- ◊ Reading address **n** or **n+1** will always latch the higher order bytes into the read-on-the-fly registers (before the optional automatic reset).
- ◊ The read-on-the-fly register ROFmid is located at a chiplet address offset of **0**, the read-on-the-fly register ROFhi is located at a chiplet address offset of **1**. For a 16 bit counter the **most** significant byte should be read from ROFmid. For a 24 bit counter the **most** significant byte is read from ROFhi and the **middle** significant byte from ROFmid.
- ◊ To completely read a 24 bit counter: first read least significant byte from counter address offset **n** or **n+1**, followed by reading the ROFmid (address offset **0**) and ROFhi (address offset **1**) registers [18] .
- ◊ Every counter has an enable bit in the counter enable register (chiplet address offset **2** or **3**). This configuration bit (read/write) acts as an ON/OFF switch of the counter.
- ◊ The counters optionally have up to two programmable thresholds. These registers are located at the address offsets immediately following the counter addresses.
- ◊ Associated with each counter are a maskable interrupt bit indicating overflow and up to two maskable interrupt bits indicating threshold overstep. These bits are stored in the chiplet's pending interrupt request registers[19] .

The address allocation of each chiplet follows the same pattern as shown in Table 7.2.

[18] In this sequence.

[19] More information about interrupts can be found in chapter 7.4.

Offset	Access	Register
0	R	(Optional) common read-on-the-fly register (middle significant byte)
1	R	(Optional) common read-on-the-fly register (most significant byte)
2	R/W	Counter enable configuration register
3	R/W	(Optional) counter enable register
--	--	---
n	R	Counter register (least significant byte); **no effect** on counter
n+1	R	Counter register (least significant byte); counter is **reset**
n+2	R/W	(Optional) threshold register (least significant byte)
n+3	R/W	(Optional) threshold register (more significant byte)
--	--	---

Table 7.2 Address allocation for counters (hexadecimal offset to chiplet address)

In Figure 7.9 the timing diagram for reading a 16bit counter is shown. Simultaneously, the counter is reset to zero (read cycle from counter offset address **n+1**).

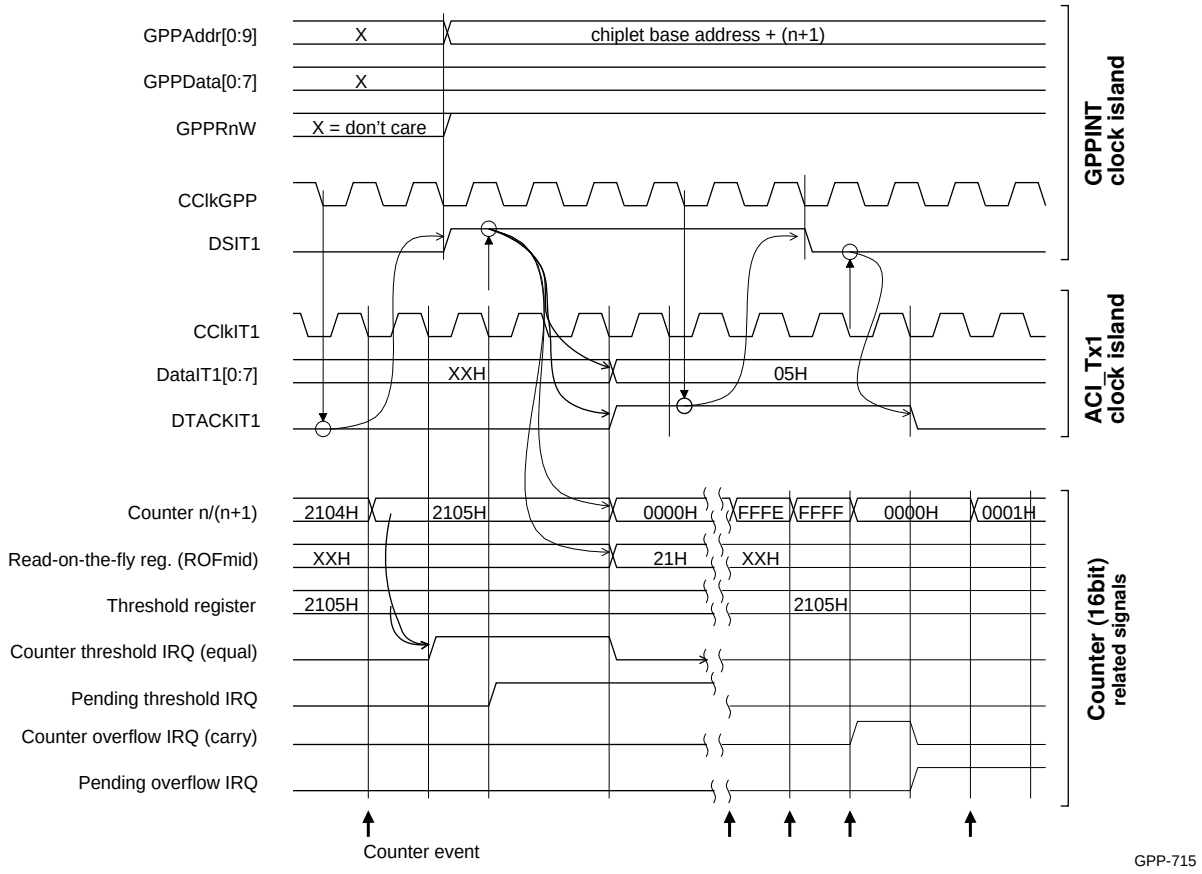

Figure 7.9: Read LSByte, latch and reset 16bit counter (example, one chiplet only)

The figure further depicts an example of the timing for the interrupt request sources of the counter (threshold overstep, counter wrap around/overflow):

- the counter threshold interrupt request (interrupt cause) is asserted for one clock cycle after the counter value is equal to the threshold value. The corresponding pending threshold interrupt request must be reset as described in chapter 7.4 (not shown in this example).
- The counter wrap around interrupt request (interrupt cause) is asserted for one clock cycle in case of counter wrap around (counter carry active). The corresponding pending threshold interrupt request must be reset as described in chapter 7.4 (not shown in this example).

7.6 Configuration, Command and Status Registers

Each chiplet holds additional status information in dedicated **status** registers. These registers are described for each chiplet in individual documents. The bits of status registers can be read-only or read/write[20]. A read-only status bit reflects always the actual state of the corresponding cause. Manipulating the cause (e.g. by command registers) automatically sets the corresponding status bit accordingly.

A read/write status bit latches and indicates the occurrence of a single event[21] which may not stay asserted until it is recognized by the microprocessor. Two mechanisms are provided to reset read/write status bits (event latch register bits):

- ◊ Reading the status register automatically resets all read/write bits (set to zero).
- ◊ Performing a write operation into the status register with a specific reset pattern. The control logic however does not simply overwrite the register contents with this reset pattern. For each bit position set in the reset pattern, the corresponding read/write bit position in the status register is reset (set to zero). This avoids unintended loss of status information.

A reset can be always performed by the write operation. The additional automatic reset upon reading the register can be enabled/disabled by a non-static configuration signal.

Bits (signals) of **configuration** registers are used to control the chiplet (FSM, logic, data manipulation) in a semi-permanent manner and are set/reset by the microprocessor. They do not change until they are again modified by the microprocessor. Configuration registers are always read/write.

Bit (signals) of **command** registers are used to control the chiplet (FSM, logic, data manipulation) in a single-shot manner. When a command bit is set high by the microprocessor (write cycle), a dedicated action is executed by the chiplet. The command register bit remains high for one chiplet clock cycle and is then automatically reset. Command registers are read/write, but reading will always result in a value of zero.

7.7 Internal Interface Ports

All signals of the GPPINT chip-internal interfaces for the transfer of non-static configuration and status information are given in Table 7.3 and 7.4. The column *Signal name* lists the name of the port signals of the GPPINT module. The column *Width* indicates the bus-width of the signal. A single bit signal has a width of 1. A data bus has typically a width of 8 (indicated by [0:7]). Index 0 is the most significant bit. The column *Default* indicates the default value of the signal after reset.

Note:

- ◊ The CS_Tx, CS_Rx and LIM (PIM and SIM) chiplets only require the GPPINT reset signal(s) and clock monitoring signals. All other configuration is done via static configuration signals. These chiplets do not have any internal counters or additional configuration/status registers.
- ◊ The OFP_Tx1/2/3/4, ACH_Tx1/2/3/4 and ACH_Rx1/2/3/4 interfaces belong to one clock island respectively.
- ◊ As described in detail in chapter 7.3, the GPPINT provides a continuous monitoring of all other chiplet's clocks. In case of multiple interfaces to one clock island like OFP_Tx, ACH_Tx and ACH_Rx, only one interface provides the monitoring function. Thus, OFP_Tx2/3/4 and ACH_Rx interfaces do not output a CAckClk signal to the GPPINT.

---

[20] In the SMART TESTSITE chiplets, only read/write status bits (event latch bits) are used !

[21] Once the read/write status bit (event latch bit) is set, further events have no further effect.

Signal names	Width	Default	Destination	Description
GPPData[0:7]	8	00H	ACI_Tx1, ACI_Rx1, ACI_Tx2, ACI_Rx2, PH_Tx1...4, PH_Rx1...4, ACH_Tx1...4, ACH_Rx1...4, OFP_Tx1...4, OFP_Rx1...4	Data bus to write configuration data into the chiplets. GPPData[0] is most significant bit.
GPPAddr[0:9]	10	00H	ACI_Tx1, ACI_Rx1, ACI_Tx2, ACI_Rx2, PH_Tx1...4, PH_Rx1...4, ACH_Tx1...4, ACH_Rx1...4, OFP_Tx1...4, OFP_Rx1...4	Address bus to address a dedicated register or GRA location in a chiplet for a read or write cycle. GPPAddr[0] is most significant bit.
GPPRnW	1	1	ACI_Tx1, ACI_Rx1, ACI_Tx2, ACI_Rx2, PH_Tx1...4, PH_Rx1...4, ACH_Tx1...4, ACH_Rx1...4, OFP_Tx1...4, OFP_Rx1...4	Control signal defining the direction of the data transfer: 1: Read cycle, data transfer from the chiplet to the GPPINT. 0: Write cycle, data transfer from the GPPINT to the chiplet.
ClkTest	1	0	ACI_Tx1, ACI_Rx1, ACI_Tx2, ACI_Rx2, PH_Tx1...4, PH_Rx1...4, CS_Tx, CS_Rx, ACH_Tx1, OFP_Tx1, OFP_Rx1...4	Test signal to monitor the chiplet clocks. It is periodically toggled by the GPPINT. There is only one test signal for all clock islands.
DSIT1	1	0	ACI_Tx1	Control signal from the GPPINT to the chiplet requesting a data transfer: 1: A data transfer is requested. 0: A data transfer is not requested.
ResIT1	1	1	ACI_Tx1	Reset signal from the GPPINT to the chiplet: 1: Chiplet is reset. 0: Chiplet is not reset.
DSIT2	1	0	ACI_Tx2	Same as DSIT1.
DSIR1	1	0	ACI_Rx1	Same as DSIT1.
DSIR2	1	0	ACI_Rx2	Same as DSIT1.
DSPT1	1	0	PH_Tx1	Same as DSIT1.
DSPT2	1	0	PH_Tx2	Same as DSIT1.
DSPT3	1	0	PH_Tx3	Same as DSIT1.
DSPT4	1	0	PH_Tx4	Same as DSIT1.

Table 7.3: GPPINT internal interface output signals part 1 of 3

Signal names	Width	Default	Destination	Description
DSPR1	1	0	PH_Rx1	Same as DSIT1.
DSPR2	1	0	PH_Rx2	Same as DSIT1.
DSPR3	1	0	PH_Rx3	Same as DSIT1.
DSPR4	1	0	PH_Rx4	Same as DSIT1.
DSHT1	1	0	ACH_Tx1	Same as DSIT1.
DSHT2	1	0	ACH_Tx2	Same as DSIT1.
DSHT3	1	0	ACH_Tx3	Same as DSIT1.
DSHT4	1	0	ACH_Tx4	Same as DSIT1.
DSHR1	1	0	ACH_Rx1	Same as DSIT1.
DSHR2	1	0	ACH_Rx2	Same as DSIT1.
DSHR3	1	0	ACH_Rx3	Same as DSIT1.
DSHR4	1	0	ACH_Rx4	Same as DSIT1.
DSOT1	1	0	OFP_Tx1	Same as DSIT1.
DSOT2	1	0	OFP_Tx2	Same as DSIT1.
DSOT3	1	0	OFP_Tx3	Same as DSIT1.
DSOT4	1	0	OFP_Tx4	Same as DSIT1.
DSOR1	1	0	OFP_Rx1	Same as DSIT1.
DSOR2	1	0	OFP_Rx2	Same as DSIT1.
DSOR3	1	0	OFP_Rx3	Same as DSIT1.
DSOR4	1	0	OFP_Rx4	Same as DSIT1.
ResIT2	1	1	ACI_Tx2	Same as ResIT1.
ResIR1	1	1	ACI_Rx1	Same as ResIT1.
ResIR2	1	1	ACI_Rx2	Same as ResIT1.
ResCT	1	1	CS_Tx	Same as ResIT1.
ResCR	1	1	CS_Rx	Same as ResIT1.
ResPT1	1	1	PH_Tx1	Same as ResIT1.
ResPT2	1	1	PH_Tx2	Same as ResIT1.
ResPT3	1	1	PH_Tx3	Same as ResIT1.
ResPT4	1	1	PH_Tx4	Same as ResIT1.
ResPR1	1	1	PH_Rx1	Same as ResIT1.
ResPR2	1	1	PH_Rx2	Same as ResIT1.
ResPR3	1	1	PH_Rx3	Same as ResIT1.
ResPR4	1	1	PH_Rx4	Same as ResIT1.
ResHT1	1	1	ACH_Tx1	Same as ResIT1.
ResHT2	1	1	ACH_Tx2	Same as ResIT1.
ResHT3	1	1	ACH_Tx3	Same as ResIT1.
ResHT4	1	1	ACH_Tx4	Same as ResIT1.
ResHR1	1	1	ACH_Rx1	Same as ResIT1.
ResHR2	1	1	ACH_Rx2	Same as ResIT1.
ResHR3	1	1	ACH_Rx3	Same as ResIT1.
ResHR4	1	1	ACH_Rx4	Same as ResIT1.
ResOT1	1	1	OFP_Tx1	Same as ResIT1.
ResOT2	1	1	OFP_Tx2	Same as ResIT1.
ResOT3	1	1	OFP_Tx3	Same as ResIT1.
ResOT4	1	1	OFP_Tx4	Same as ResIT1.
ResOR1	1	1	OFP_Rx1	Same as ResIT1.
ResOR2	1	1	OFP_Rx2	Same as ResIT1.
ResOR3	1	1	OFP_Rx3	Same as ResIT1.
ResOR4	1	1	OFP_Rx4	Same as ResIT1.

Table 7.3: GPPINT internal interface output signals part 2 of 3

Signal names	Width	Default	Destination	Description
TxRPIM	1	1	PIM_Tx	Transmit Reset Parallel Interface Module.
TxRPIS1	1	1	PIS_Tx1	Transmit Reset Parallel Interface Section #1.
TxRPIS2	1	1	PIS_Tx2	Transmit Reset Parallel Interface Section #2.
TxRPIS3	1	1	PIS_Tx3	Transmit Reset Parallel Interface Section #3.
TxRPIS4	1	1	PIS_Tx4	Transmit Reset Parallel Interface Section #4.
RxRPIM	1	1	PIM_Rx	Receive Reset Parallel Interface Module.
RxRPIS1	1	1	PIS_Rx1	Receive Reset Parallel Interface Section #1.
RxRPIS2	1	1	PIS_Rx2	Receive Reset Parallel Interface Section #2.
RxRPIS3	1	1	PIS_Rx3	Receive Reset Parallel Interface Section #3.
RxRPIS4	1	1	PIS_Rx4	Receive Reset Parallel Interface Section #4.
APSTxReset	1	1	ACH_Tx	Reset to APS_Tx
APSRxReset	1	1	ACH_Rx	Reset to APS_Rx
TxReset	1	1	SIM	Reset to SIM Transmit Section
RxReset	1	1	SIM	Reset to SIM Receive Section
TxReset1	1	1	SIM	Reset to SIM Transmit Section #1
TxReset2	1	1	SIM	Reset to SIM Transmit Section #2
TxReset3	1	1	SIM	Reset to SIM Transmit Section #3
TxReset4	1	1	SIM	Reset to SIM Transmit Section #4
RxReset1	1	1	SIM	Reset to SIM Receive Section #1
RxReset2	1	1	SIM	Reset to SIM Receive Section #2
RxReset3	1	1	SIM	Reset to SIM Receive Section #3
RxReset4	1	1	SIM	Reset to SIM Receive Section #4

Table 7.3: GPPINT internal interface output signals part 3 of 3

Signal names	Width	Source	Default	Description
DataIT1[0:7] DataIT2[0:7]	8 8	ACI_Tx1 ACI_Tx2	00H 00H	Data bus to read configuration or status information out of the chiplets. Index [0] is most significant bit.
DTACKIT1 DTACKIT2	1 1	ACI_Tx1 ACI_Tx2	0 0	Data transfer acknowledge: 1: Read cycle: the chiplet outputs valid data on its outgoing data bus. Write cycle: the chiplet has read and latched the data received on GPPData[0:7]. 0: Data on the outgoing data bus is not valid, data on the incoming GPPData data bus has not yet been accepted.
IRQIT1 IRQIT2	1 1	ACI_Tx1 ACI_Tx2	0 0	Interrupt request: 1: The chiplet requests to generate an interrupt towards the microprocessor. 0: The chiplet does not request an interrupt. The cause of the request can be read out of the pending interrupt request register inside the chiplet.
CAckIT1 CAckIT2	1 1	ACI_Tx1 ACI_Tx2	-- --	Acknowledge signal for clock monitoring. It is not set to a default value by reset.
DataIR1[0:7] DataIR2[0:7]	8 8	ACI_Rx1 ACI_Rx2	00H 00H	Same as DataIT1.
DTACKIR1 DTACKIR2	1 1	ACI_Rx1 ACI_Rx2	0 0	Same as DTACKIT1.
IRQIR1 IRQIR2	1 1	ACI_Rx1 ACI_Rx2	0 0	Same as IRQIT1.
CAckIR1 CAckIR2	1 1	ACI_Rx1 ACI_Rx2	-- --	Same as CAckIT1.
DataPT1[0:7] DataPT2[0:7] DataPT3[0:7] DataPT4[0:7]	8 8 8 8	PH_Tx1 PH_Tx2 PH_Tx3 PH_Tx4	00H 00H 00H 00H	Same as DataIT1.
DTACKPT1 DTACKPT2 DTACKPT3 DTACKPT4	1 1 1 1	PH_Tx1 PH_Tx2 PH_Tx3 PH_Tx4	0 0 0 0	Same as DTACKIT1.
IRQPT1 IRQPT2 IRQPT3 IRQPT4	1 1 1 1	PH_Tx1 PH_Tx2 PH_Tx3 PH_Tx4	0 0 0 0	Same as IRQIT1.
CAckPT1 CAckPT2 CAckPT3 CAckPT4	1 1 1 1	PH_Tx1 PH_Tx2 PH_Tx3 PH_Tx4	-- -- -- --	Same as CAckIT1.

Table 7.4: GPPINT internal interface input signals part 1 of 3

Signal names	Width	Source	Default	Description
DataPR1[0:7]	8	PH_Rx1	00H	Same as DataIT1.
DataPR2[0:7]	8	PH_Rx2	00H	
DataPR3[0:7]	8	PH_Rx3	00H	
DataPR4[0:7]	8	PH_Rx4	00H	
DTACKPR1	1	PH_Rx1	0	Same as DTACKIT1.
DTACKPR2	1	PH_Rx2	0	
DTACKPR3	1	PH_Rx3	0	
DTACKPR4	1	PH_Rx4	0	
IRQPR1	1	PH_Rx1	0	Same as IRQIT1.
IRQPR2	1	PH_Rx2	0	
IRQPR3	1	PH_Rx3	0	
IRQPR4	1	PH_Rx4	0	
CAckPR1	1	PH_Rx1	--	Same as CAckIT1.
CAckPR2	1	PH_Rx2	--	
CAckPR3	1	PH_Rx3	--	
CAckPR4	1	PH_Rx4	--	
DataHR1[0:7]	8	ACH_Rx1	00H	Same as DataIT1.
DataHR2[0:7]	8	ACH_Rx2	00H	
DataHR3[0:7]	8	ACH_Rx3	00H	
DataHR4[0:7]	8	ACH_Rx4	00H	
DTACKHR1	1	ACH_Rx1	0	Same as DTACKIT1.
DTACKHR2	1	ACH_Rx2	0	
DTACKHR3	1	ACH_Rx3	0	
DTACKHR4	1	ACH_Rx4	0	
IRQHR1	1	ACH_Rx1	0	Same as IRQIT1.
IRQHR2	1	ACH_Rx2	0	
IRQHR3	1	ACH_Rx3	0	
IRQHR4	1	ACH_Rx4	0	
DataHT1[0:7]	8	ACH_Tx1	00H	Same as DataIT1.
DataHT2[0:7]	8	ACH_Tx2	00H	
DataHT3[0:7]	8	ACH_Tx3	00H	
DataHT4[0:7]	8	ACH_Tx4	00H	
DTACKHT1	1	ACH_Tx1	0	Same as DTACKIT1.
DTACKHT2	1	ACH_Tx2	0	
DTACKHT3	1	ACH_Tx3	0	
DTACKHT4	1	ACH_Tx4	0	

Table 7.4: GPPINT internal interface input signals part 2 of 3

Signal names	Width	Source	Default	Description
IRQHT1	1	ACH_Tx1	0	Same as IRQIT1.
IRQHT2	1	ACH_Tx2	0	
IRQHT3	1	ACH_Tx3	0	
IRQHT4	1	ACH_Tx4	0	
CAckHT1	1	ACH_Tx	--	Same as CAckIT1.
DataOT1[0:7]	8	OFP_Tx1	00H	Same as DataIT1.
DataOT2[0:7]	8	OFP_Tx2	00H	
DataOT3[0:7]	8	OFP_Tx3	00H	
DataOT4[0:7]	8	OFP_Tx4	00H	
DTACKOT1	1	OFP_Tx1	0	Same as DTACKIT1.
DTACKOT2	1	OFP_Tx2	0	
DTACKOT3	1	OFP_Tx3	0	
DTACKOT4	1	OFP_Tx4	0	
IRQOT1	1	OFP_Tx1	0	Same as IRQIT1.
IRQOT2	1	OFP_Tx2	0	
IRQOT3	1	OFP_Tx3	0	
IRQOT4	1	OFP_Tx4	0	
CAckOT1	1	OFP_Tx1	--	Same as CAckIT1.
DataOR1[0:7]	8	OFP_Rx1	00H	Same as DataIT1.
DataOR2[0:7]	8	OFP_Rx2	00H	
DataOR3[0:7]	8	OFP_Rx3	00H	
DataOR4[0:7]	8	OFP_Rx4	00H	
DTACKOR1	1	OFP_Rx1	0	Same as DTACKIT1.
DTACKOR2	1	OFP_Rx2	0	
DTACKOR3	1	OFP_Rx3	0	
DTACKOR4	1	OFP_Rx4	0	
IRQOR1	1	OFP_Rx1	0	Same as IRQIT1.
IRQOR2	1	OFP_Rx2	0	
IRQOR3	1	OFP_Rx3	0	
IRQOR4	1	OFP_Rx4	0	
CAckOR1	1	OFP_Rx1	--	Same as CAckIT1.
CAckOR2	1	OFP_Rx2	--	
CAckOR3	1	OFP_Rx3	--	
CAckOR4	1	OFP_Rx4	--	
CAckCT	1	CS_Tx	--	Same as CAckIT1.
CAckCR	1	CS_Rx	--	

Table 7.4: GPPINT internal interface input signals part 3 of 3

8 Chiplet Address Mapping

Table 8.1 lists the address mapping of all chiplets of SMART[22].

Chiplet base address	Chiplet address range	# of bytes	Chiplet	(short name)
0000 h	0000 - 000F h	16	Access protection mechanism (GPPINT)	GPPINT
---	0010 - 00FF h	240	reserved	---
0100 h	0100 - 01FF h	256	ACI_Tx1	IT1
0200 h	0200 - 02FF h	256	ACI_Tx2	IT2
0300 h	0300 - 03FF h	256	ACI_Rx1	IR1
0400 h	0400 - 04FF h	256	ACI_Rx2	IR2
0500 h	0500 - 05FF h	256	PH_Tx1	PT1
0600 h	0600 - 06FF h	256	PH_Tx2	PT2
0700 h	0700 - 07FF h	256	PH_Tx3	PT3
0800 h	0800 - 08FF h	256	PH_Tx4	PT4
0900 h	0900 - 09FF h	256	PH_Rx1	PR1
0A00 h	0A00 - 0AFF h	256	PH_Rx2	PR2
0B00 h	0B00 - 0BFF h	256	PH_Rx3	PR3
0C00 h	0C00 - 0CFF h	256	PH_Rx4	PR4
0D00 h	0D00 - 0DFF h	256	ACH_Tx1	HT1
0E00 h	0E00 - 0EFF h	256	ACH_Tx2	HT2
0F00 h	0F00 - 0FFF h	256	ACH_Tx3	HT3
1000 h	1000 - 10FF h	256	ACH_Tx4	HT4
1100 h	1100 - 11FF h	256	ACH_Rx1	HR1
1200 h	1200 - 12FF h	256	ACH_Rx2	HR2
1300 h	1300 - 13FF h	256	ACH_Rx3	HR3
1400 h	1400 - 14FF h	256	ACH_Rx4	HR4
1500 h	1500 - 17FF h	768	reserved	---
1800 h	1800 - 1BFF h	1024	OFP_Tx1	OT1
1C00 h	1C00 - 1FFF h	1024	OFP_Tx2	OT2
2000 h	2000 - 23FF h	1024	OFP_Tx3	OT3
2400 h	2400 - 27FF h	1024	OFP_Tx4	OT4
2800 h	2800 - 2BFF h	1024	OFP_Rx1	OR1
2C00 h	2C00 - 2FFF h	1024	OFP_Rx2	OR2
3000 h	3000 - 33FF h	1024	OFP_Rx3	OR3
3400 h	3400 - 37FF h	1024	OFP_Rx4	OR4
---	---	---	CS_Tx	CT
---	---	---	CS_Rx	CR
3800 h	3800 - 38FF h	256	GPPINT	GPPINT
3900 h	3900 - 3FAF h	1712	reserved	---
3FB0 h	3FB0 - 3FBF h	16	Access protection mechanism (GPPINT)	GPPINT
3FC0 h	3FC0 - 3FFF h	64	reserved (Utopia Level 2 Pointers)	GPPINT

Table 8.1: Chiplet address mapping

[22] CS_Tx and CS_Rx are controlled by static configuration signals and do not require an individual address space.

9 External Microprocessor Interface

The external microprocessor interface, controlled by the GPPINT module, is a QUICC bus type interface [9, 10]. The GPPCLK frequency range is specified between 10 MHz and 50 MHz. Synchronous or asynchronous mode of operation can be selected by an external input setting (pin IntfMode). In synchronous mode of operation the GPPCLK clock is in phase with the processor clock. The interface was designed for the MC68360 microprocessor with highest priority. It is adaptable to other processor types that allow unlimited wait state insertion, such as i960C and i80186. Other processors that only allow a limited number of wait states, such as PowerPC403, can be adapted with some restrictions to the chiplet clock frequencies.

All Inputs/Outputs of the microprocessor interface are 5.0V TTL compatible. Drivers are 50Ω source-terminated. The GPDTACK output can be in all 3 states in operational mode (H,L,Z) and needs a pullup resistor on the system bus. The SMARTInt IRQ output is in the high-impedance state (Z) when deasserted. A typical interface circuit is shown in Figure 9.1. More glue logic may be required for high frequency (>25 MHz) synchronous operation. Additional circuitry is needed depending on processor type, bus buffering, board configuration etc. .

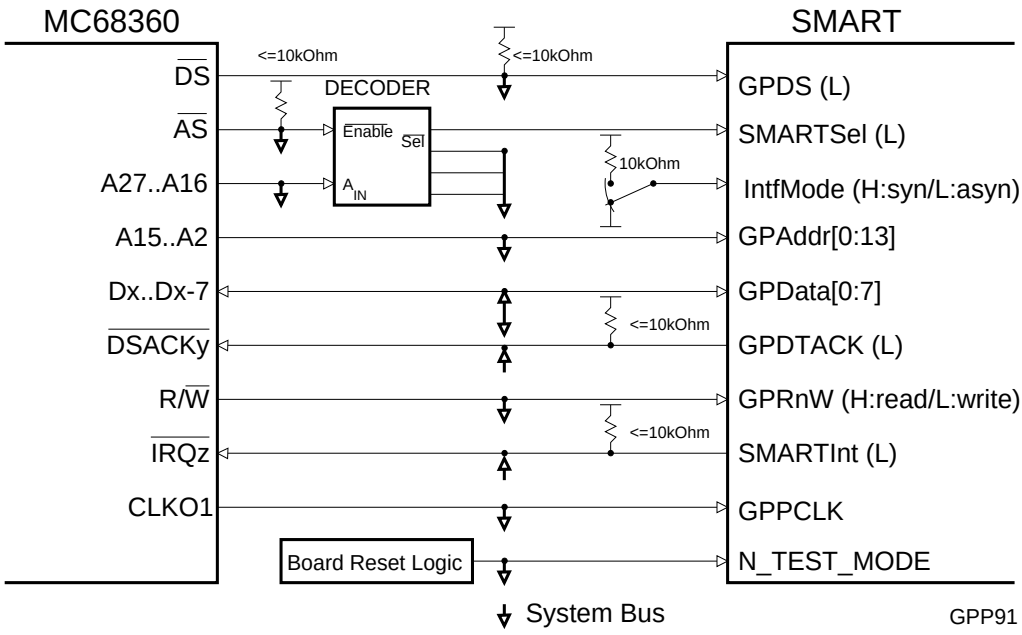

Figure 9.1: Interfacing SMART to MC68360 microprocessor; x=7,15,23,31 , y=0,1 and z=1..7 depending on system data configuration and addressing scheme used.

The timing at the interface to a QUICC bus type processor is shown in Figure 9.2 for the **synchronous** mode of operation and in Figure 9.3 for the **asynchronous** mode of operation.

In **synchronous** mode (IntfMode=1), there is always at least one wait state cycle of the microprocessor. For both read and write accesses, wait cycles have to be added if the previous access has not been completed SMART-internally. This is only important if the access cycles follow each other at short intervals. Additional wait states are required for read accesses from SMART chiplets; the lower the

frequency of the chiplet clock, the longer it will take until the chiplet data is ready on the external bus. The synchronous write access interaction between the QUICC processor and the GPPINT module of SMART is as follows (see Figure 9.2):

1) After updating the GPAddr bus and deasserting GPRnW (L) the microprocessor asserts SMARTSel (L); the GPDTACK signal is deasserted (H) by SMART.
2) The microprocessor asserts GPDS (L) and enables its GPData bus drivers. If SMART has already completed the previous access internally, GPDTACK is asserted immediately (L).
3) If SMART is still busy with the previous write access internally, the assertion of GPDTACK is delayed and the microprocessor inserts wait cycles.
4) At the next falling edge of GPPCLK following the assertion of GPDTACK the address and data signals are latched into the SMART address and command registers. With the same clock edge the microprocessor detects GPDTACK "low". One clock period later the microprocessor deasserts SMARTSel (H) and GPDS (H); at the same time it disables its GPData bus drivers.
5) Following deassertion of SMARTSel or GPDS (whatever occurs first) the SMART chip tri-states GPDTACK (Z).

The read cycle interaction is as follows:

6) After updating the GPAddr bus and asserting GPRnW (H) the microprocessor asserts SMARTSel (L) and GPDS (L); the GPDTACK signal is deasserted (H) by SMART.
7) If SMART has already completed the previous access internally, the address is latched immediately by SMART, otherwise wait cycles are inserted by SMART before latching the address. After address latching the SMART chip forces more wait cycles until the internal (chiplet) data is ready.
8) When the internal data is ready SMART asserts GPDTACK (L), puts valid data onto the GPData bus and enables the bus drivers.
9) The microprocessor detects GPDTACK "low" with the next falling edge of GPPCLK. The SMART data is latched into the microprocessor one cycle later.
10) Just after latching the data, the microprocessor deasserts SMARTSel (H) and GPDS (H). Following deassertion of SMARTSel or GPDS (whatever occurs first) the SMART chip tri-states GPDTACK (Z) and disables the GPData bus drivers.

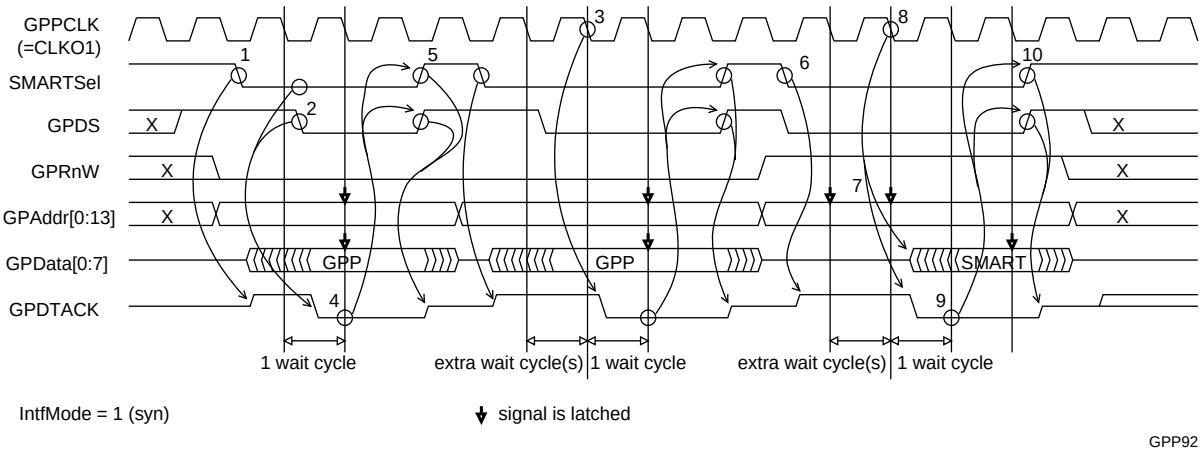

Figure 9.2: Synchronous external interface timing, showing two write cycles followed by one read access. Although 2 arrows are shown for the read access the address is latched only once; wait cycles

can be inserted before address latching (waiting for previous access to complete internally) and/or after address latching (waiting for the internal chiplet data).

In **asynchronous** mode (IntfMode=0), additional microprocessor wait cycles are required because of the necessary synchronization of the control signals in the SMART chip. The GPPINT module can even handle the situation where the microprocessor clock has a higher frequency than GPPCLK. For this case, a D-FlipFlop makes sure that the deasserted phase of the chip select (SMARTSel=1) between access cycles immediately following each other is not lost.

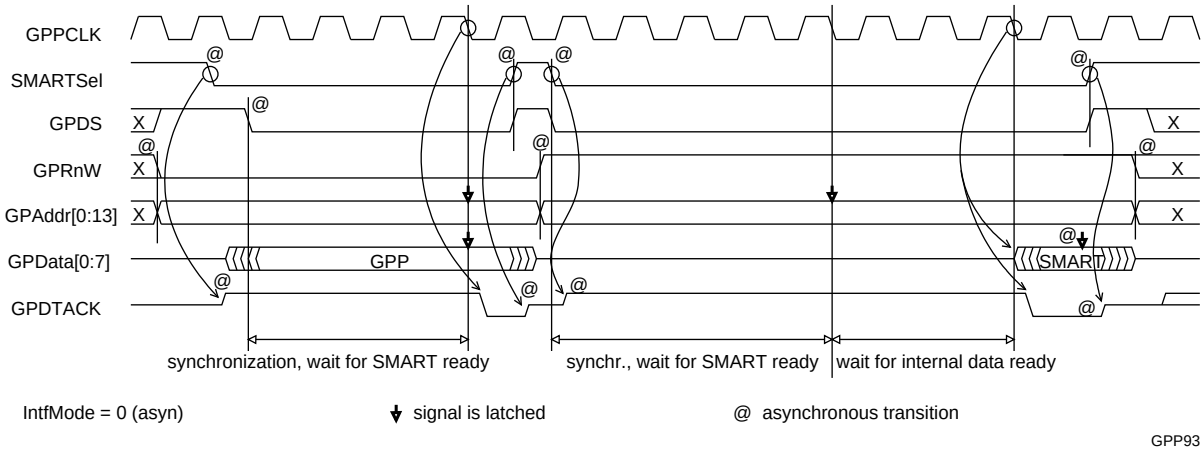

Figure 9.3: Asynchronous external interface timing, showing a write access followed by a read access. Compared to the synchronous mode, additional wait states are required for the synchronization of the control signals.

10 References

[1] "Utopia, An ATM-PHY Interface Specification", Level 1, Version 2.01, March 21, 1994, ATM Forum Recommendation.

[2] "Utopia, An ATM-PHY Interface Specification", Level 2, Version 0.8, April 10, 1995, ATM Forum Recommendation.

[3] "ITU-T G.708 (03/93), General Aspects of Digital Transmission System, Network Node Interface for the Synchronous Digital Hierarchy".

[4] "ITU-T G.709 (03/93), General Aspects of Digital Transmission System, Synchronous Multiplexing Structure", chapter 2.4, page 10.

[5] "ITU-T G.783 (01/94), General Aspects of Digital Transmission System, Characteristics of Synchronous Digital Hierarchy (SDH) Equipment Functional Blocks ".

[6] "ITU-T G.803 (03/93), Digital Networks, Architectures of Transport Networks Based on the Synchronous Digital Hierarchy (SDH)".

[7] "Special Report, SR_NWT-001756", Bellcore .

[8] "Scrambling Techniques for Digital Transmission", Byeong Gi Lee, Seok Chang Kim, Springer Verlag, 1994, ISBN 3-540-19863-6, ISBN 0-387-19863-6.

[9] "MC68360, Quad Integrated Communications Controller", User's Manual, Motorola, MC68369UM/AD, 3/93.

[10] "Errata and Added Information to MC68360, Quad Integrated Communications Controller User's Manual", Motorola, MC68369UMAD/AD, 8/93.

[11] "SMART: Line Interface Description ", Wolfram Lemppenau (wle @ zurich.ibm.com), IBM Zurich Research Laboratory, 12/11/95, Version 2.2.

End of document