ARSHXIP User Manual

Features

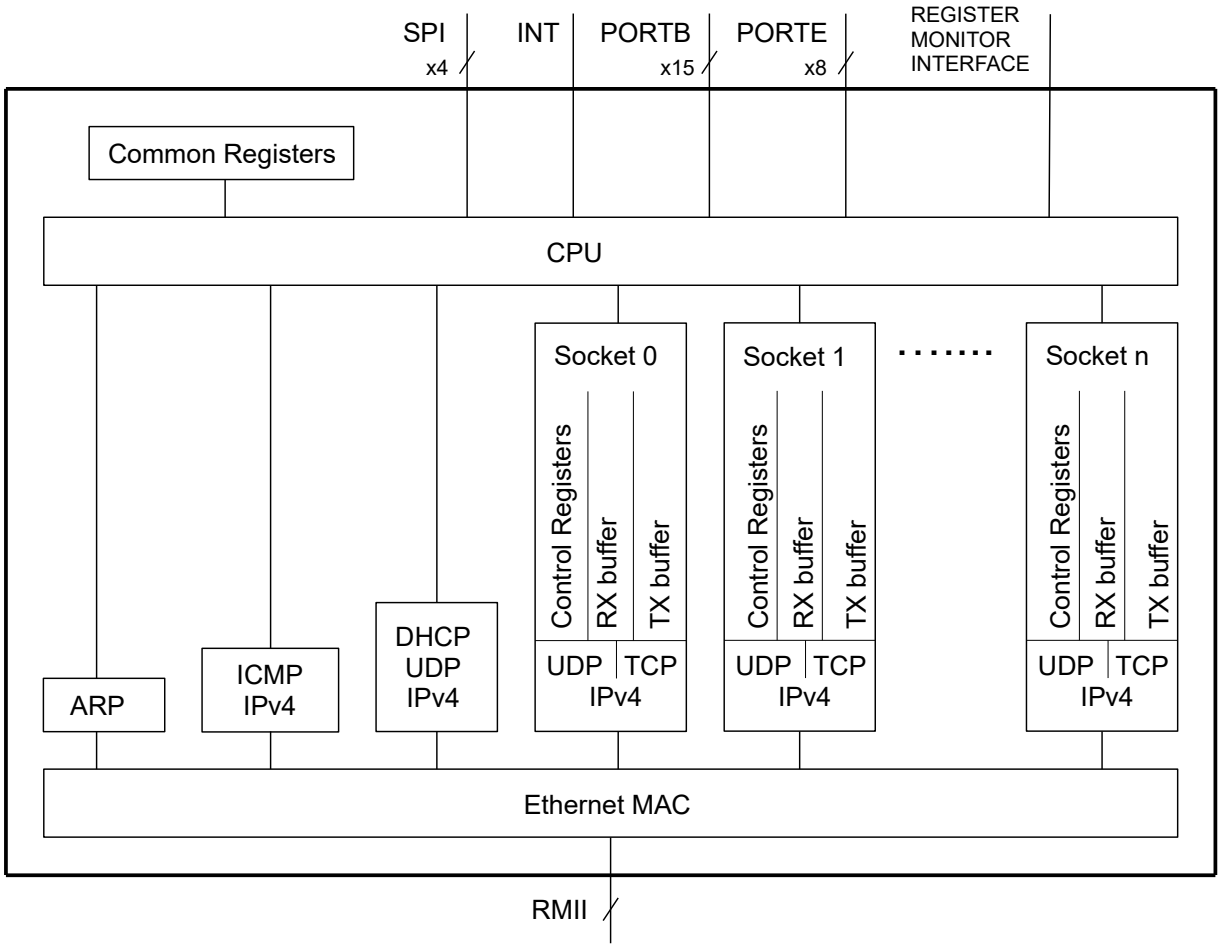
- PIC32MX664F064H microcontroller with unique MAC address
- 20 kB flexible allocation receive / transmit total buffer memory
- 4 multi-subnet independent sockets
- one 8 bit GPIO port 5V tolerant
- one 15 bit GPIO port (3.3V)
- 20MHz SPI mode 0, 1, 2 or 3
- protocols: ARP, IPv4, ICMP, UDP, TCP, DHCP
- PING reply algorithm
- Interrupt System
- UART Register Monitor Interface
- 5V to 3.3V linear regulator

Table of Contents

1. Overview

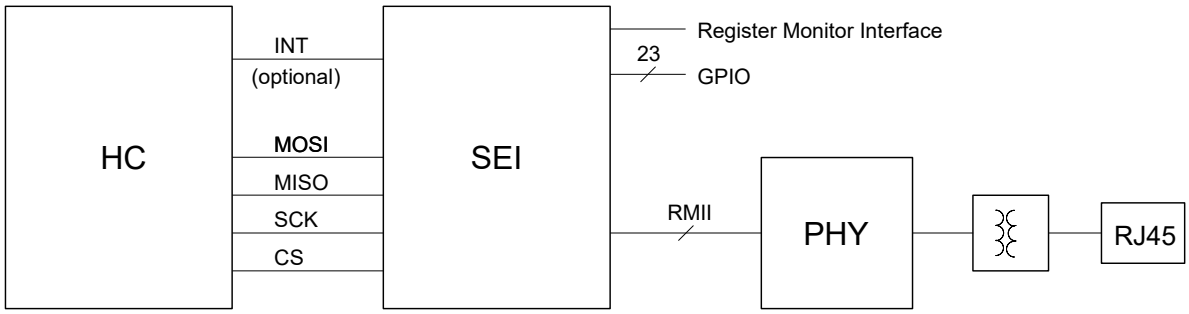
ARSHXIP is part of an SPI to Ethernet Interface family of products. These products take a form of an Ethernet Shield meant to work with Arduino Uno, Mega etc. boards. ARSHXIP is built around a PIC32MX664F064H microcontroller (referenced in this document as PIC32) programmed with a TCP-IP stack. PIC32 connects on one side to a host controller – referenced here as HC – via an SPI interface and on the other side to a PHY device device via an RMII interface. It does not run user firmware. The logic, firmware or software that drives this product is referenced in this document as “SEI” – SPI to Ethernet Interface.

Fig. 1-1: SEI block diagram



1.1 Hardware Implementation

Fig.1.1-1: Block diagram of a system with SEI



HC sends data to PIC32 over the SPI interface. PIC32 wraps data in a packet and sends it to a PHY device over the RMII interface. The PHY device then sends the packet on the network. When receiving data the path is from the network, PHY, PIC32. PIC32 buffers data until HC reads it over the SPI interface.

For the input clock, PIC32 uses the same 50MHz clock it needs for the MAC controller, which is provided by the PHY device attached to it.

PIC32 provides an output pin to be connected to the PHY's RESET pin. This pin can be controlled via the SPI interface.

2 Memory allocation

HC interacts with the PIC32 by reading / writing to various registers, and reading / writing payload data to a number of circular buffers.

HC handles the UDP and TCP protocols via sockets. 4 identical sockets are implemented. In this context, a socket represents a set of registers, a receive buffer, a transmit buffer and a routine that gets serviced by the CPU. A total of 20480 bytes in PIC32 RAM memory is available for receive / transmit socket buffers.

Registers used to handle the sockets are called socket registers. Registers used to handle other protocols like ICMP, DHCP etc and other functionalities are called common registers. They are not mapped to specific addresses. The HC reads and writes them via specific SPI commands. Some of the registers are combined in one SPI command. This arrangement aims to reduce the number of SPI commands which is limited, and reduce overhead.

When not all sockets are used, the respective buffers can be reallocated to those used. Buffer memory allocation is made in units of 2048 bytes. A receive or transmit buffer must be continuous and not to overlap with other buffers. These conditions must be followed by HC when reallocating buffers.

Table 2-2: Common registers summary

Name	Comment	Size [bytes]	Read / Write
GC	General Control	2	R/W
GS	General Status	2	R
SMR	SPI Mode and Reset	2	W
HMA	Host Mac Address	6	R/W
HIPA	Host IP Address	4	R/W
HSM	Host Subnet Mask	4	R/W
GIPA	Gateway IP Address	4	R/W
EPR	Ephemeral Port Range	4	R/W
IF	Interrupt Flags	4	R
PHYCS	PHY Control and Status	2	R/W
PHYA	PHY Address	2	R/W
PHYD	PHY Data	2	R/W
ICMP	ICMP Control and Status	2	R/W
PQIP	Ping Query IP	4	R
ICMPS	ICMP packet Sent	variable	R
ICMPR	ICMP packet Received	variable	R
DHCS	DHCP Control and Status	2	R/W
DHSRV	DHCP Server IP address	4	R
DNS1	DNS Server IP address 1	4	R
DNS2	DNS Server IP address 2	4	R
DHRLT	DHCP Remaining Lease Time	4	R
DHLT	DHCP Lease Time	4	R
ARPC	ARP Control	2	R/W
ATTL	ARP TimeToLive	2	R/W
TRISB	TRISB	2	R/W
LATB	LATB	2	W

Table 2-2: Common registers summary (continuation)

Name	Comment	Size [bytes]	Read / Write
PORTB	PORTB	2	R
ODCB	ODCB	2	R/W
TRISE	TRISE	2	W
LATE	LATE	2	W
PORTE	PORTE	2	R
ODCE	ODCE	2	W
FWIDB	Firmware ID and Build	4	R

Table 2-3: Socket registers summary

Name	Comment	Size [bytes]	Read / Write
SnCS	Socket Control and Status	2	R
SnCCLR	Socket Control Clear	2	W
SnCSET	Socket Control Set	2	W
SnPMA	Peer Mac Address	6	R, W
SnIPA	IP Address	4	R, W
SnSM	Subnet Mask	4	R, W
SnPIPA	Peer IP Address	4	R, W
SnP	Socket Port	2	R, W
SnPP	Peer Port	2	R, W
SnTTL	TimeToLive	2	R/W
SnBSPS	Buffer Start Pointer and Size	4	R/W
SnRDS	Received Data Size	2/4	R
SnTFS	Transmit Buffer Free Size	2/4	R
SnMTU	Maximum Transmission Unit	2	R/W
SnPMSS	Peer Maximum Segment Size	2	R
SnTCPT	TCP Tuning	4	R/W
SnCW	Congestion Window	2	R
SnSST	Slow Start Threshold	2	R
SnRTO	Retransmit Timeout	2	R
SnSRTT	Smooth Round Trip Time	2	R
SnRHP	Receive Head Pointer	4	R
SnRTP	Receive Tail Pointer	4	R
SnTHP	Transmit Head Pointer	4	R
SnTHHP	Transmit Head Hold Pointer	4	R
SnTTP	Transmit Tail Pointer	4	R

3 SPI Interface

The SPI interface transfers data between HC and SEI. The host controller acts as master while SEI as slave. It transfers 16bits (two bytes) minimum at a time. While all registers are made up of an even number of bytes, payload data can be an even or odd number of bytes. When HC transmits an odd number of payload data, it needs to send one more (dummy) byte to complete the transaction. When it reads an odd number of payload data bytes from SEI, it needs to clock out one more byte in order to complete the transaction.

The SPI interface can be set in any of the four configurations possible. After reset, it is set in mode 3.

Hardware wise, an SPI transaction represents data exchange between HC and SEI from the moment HC activates the CS line until it deactivates it.

The SPI interface uses four lines:

- CS – chip select. The HC starts a new SPI transaction by asserting the CS line and ends it by bringing it back to the idle state. For SEI, this line is a 5V tolerant input.
- SCK – clock line driven by the HC. For SEI, this line is a 5V tolerant input.
- MOSI – Master Out Slave In. HC sends data to SEI on this line. For SEI, this line is a 5V tolerant input.
- MISO – Master In, Slave Out. SEI sends data to HC on this line. When CS line is in idle state, this line is in a high impedance state. This allows multiple slaves on the bus.

3.1 SPI Commands

From a logic point of view, HC interacts with SEI by sending various commands over the SPI interface in a form of an SPI transaction.

SEI has always two bytes in the transmit buffer waiting to be sent out. Once HC asserts the CS line, SEI places the content of GS register in the SPI transmit buffer which will be seen during byte exchange 2 and 3 (counting from 0).

If HC performs a write command, at the end of the transaction SEI places two bytes in the SPI transmit buffer to be ready to be clocked out with the next transaction. An error occurs if HC abandons the SPI transaction early. If this happen, the next transaction will be erroneous. To resynchronize with SEI, HC can simply read a two byte register and discard the result. It is important that the HC clocks in all pulses necessary for the respective transaction.

Mainly, there are two types of commands:

- type 1: commands with a fix / known payload data length. These commands are used to read / write common or socket registers. The payload data length is determined by the registers themselves
- type 2: commands with a variable payload length used to read / write UDP, TCP or other variable data length

An SPI transaction is made up of two parts: a header and payload data.

3.1.1 SPI Type 1 Commands

These commands start with a header which is two bytes long. The following table describes the header structure.

Table 3.1.1-1: SPI type 1 command header structure.

Bit offset	Size [bits]	Name		Description
0	6	CC		Command Code
6	1	RW		Read / Write. HC sets this bit when it reads data from SEI and clears it when writes data to SEI
7	1	RT		Register Type RT = 0 for socket registers RT = 1 for common registers
8	8	SN/RO	RT = 0	Socket Number. The HC places here the socket number
			RT = 1	Register Offset

Once the header is clocked in, HC continues with the payload data.

Table 3.1.1-2: Write 16 bit register timeline

HC	SEI
SPI in idle state. SEI transmit buffer has two dummy bytes in it.	
HC activates CS line	
	MISO changes state from Hi-Z to an output. GC register content is written to SPI transmit buffer.
HC starts clocking out the command header (fields CC, RW = 0, RT and SN/RO)	SEI sends out two dummy bytes
HC sends out the 16 bit register value	SEI sends out GS register value. Also during this time, it decodes the command header just received and decides if it is a valid one, or if the register can be written. If the command is not valid or the register cannot be written: - it writes two dummy bytes to the transmit buffer to be ready for the next SPI transaction - it enters a state that any subsequent data received is ignored until the CS line is activated again.
HC ends the transaction by bringing the CS line in idle state	At this point SEI received the register value to be written. Given that the command header proved right, it proceeds to update the register. In some cases a temporary register is used and the actual update is made in the main loop. SEI writes two dummy bytes to the transmit buffer to be ready for the next SPI transaction.
SPI in idle state. SEI transmit buffer has two dummy bytes in.	

Writing registers more than two bytes in size proceeds in a similar way. Once the entire new value is clocked in, the entire register is updated at once. In case of commands that can write multiple registers, each register is updated once is clocked in.

Table 3.1.1-3: Read 16 bit register timeline

HC	SEI
SPI in idle state. There are two dummy bytes in the transmit buffer.	
HC activates CS line	
	MISO changes state from Hi-Z to output. GC register content is written to the SPI transmit buffer.
HC starts clocking out the command header (fields CC, RW = 1, RT and SN/RO)	SEI sends out 0x0000 (the two dummy bytes)
HC sends out two dummy bytes	SEI sends out GS register value. Also during this time, it decodes the command header just received and decides if it is a valid one. Some registers does not actually exist, their value is calculated. For such registers, this is the time they are calculated. If the command header is right, SEI places the register content in the SPI transmit buffer If the command is not a valid one, it enters a state that any subsequent data received is ignored until the CS line is activated again.
HC sends out two dummy bytes and in the same time reads data from the MISO line.	SEI sends out the requested register value. SEI writes two dummy bytes to the transmit buffer to be ready for the next SPI transaction.
HC deactivates CS line	
SPI in idle state. There are two dummy bytes in the transmit buffer.	

Reading registers more than two bytes in size proceeds in a similar way.

3.1.2 SPI Type 2 Commands

These commands are specific to protocols like ICMP, UDP and TCP. See the respective protocol chapter for more details.

Table 3.1.2-1: SPI Commands

Cmd code	RW	RT	Register	Cmd code	RW	RT	Register
0	0-W	0	SnCSCLR	1	0-W	0	SnCSSET
		1	GC, SMR			1	HMA
	1-R	0	SnCS		1-R	0	
		1	GC			1	HMA
2	0-W	0	SnPMA	3	0-W	0	SnIPA
		1	HIPA, HSM, GIPA			1	EPR
	1-R	0	SnPMA		1-R	0	SnIPA
		1	HIPA, HSM, GIPA			1	EPR
4	0-W	0	SnSM	5	0-W	0	SnPIPA
		1				1	PHYCS
	1-R	0	SnSM		1-R	0	SnPIPA
		1	IF			1	PHYCS
6	0-W	0	SnP	7	0-W	0	SnPP
		1	PHYA			1	PHYD
	1-R	0	SnP		1-R	0	SnPP
		1	PHYA			1	PHYD
8	0-W	0	SnIETTL	9	0-W	0	SnBSPS
		1	ICMP			1	-
	1-R	0	SnTTL		1-R	0	SnBSPS
		1	ICMP			1	PQIP
10	0-W	0		11	0-W	0	
		1				1	
	1-R	0	SnRDS		1-R	0	SnTFS
		1	ICMPS			1	ICMPR
12	0-W	0	SnMTU	13	0-W	0	SnDU
		1	DHCS			1	
	1-R	0	SnMTU, SnPMSS		1-R	0	SnDU
		1	DHCS			1	DNS1, DNS2, DHSRV
14	0-W	0	SnDT	15	0-W	0	SnTCPT
		1				1	ARPC
	1-R	0	SnDT		1-R	0	SnTCPT
		1	DHRLT, DHLT			1	ARPC

Table 3.1.2-1: SPI Commands (continuation)

Cmd code	RW	RT	Register	Cmd code	RW	RT	Register
16	0-W	0		17	0-W	0	
		1	ATTL			1	
	1-R	0			1-R	0	
		1	ATTL			1	
20	0-W	0		21	0-W	0	
		1	TRISB			1	LATB
	1-R	0			1-R	0	
		1	TRISB			1	PORTB
22	0-W	0		23	0-W	0	
		1	ODCB			1	TRISE
	1-R	0			1-R	0	
		1	ODCB			1	TRISE
24	0-W	0		25	0-W	0	
		1	LATE			1	ODCE
	1-R	0			1-R	0	
		1	PORTE			1	ODCE
54	0-W	0		55	0-W	0	
		1				1	
	1-R	0			1-R	0	SnCW
		1				1	
56	0-W	0		57	0-W	0	
		1				1	
	1-R	0	SnSST		1-R	0	SnRTO
		1				1	
58	0-W	0		59	0-W	0	
		1				1	
	1-R	0	SnSRTT		1-R	0	SnRHP
		1				1	
60	0-W	0		61	0-W	0	
		1				1	
	1-R	0	SnRTP		1-R	0	SnTHP
		1				1	
62	0-W	0		63	0-W	0	
		1				1	
	1-R	0	SnTHHP		1-R	0	SnTTP
		1				1	FWIDB

4 PHY – Physical Layer Interface

4.1 PHY Overview

SEI uses an external device for the physical layer – called here “PHY”.

SEI connects to IP101GR via the RMII interface.

On power on, SEI configures the PHY in Auto Negotiation mode. Once configured, it reads the link status from PHY every 5ms. This is reflected in bit LKUP (bit 5) in GS (General Status) register which HC reads with every SPI transaction.

Before sending an Ethernet frame, SEI checks the LKUP bit. If link is down, the Ethernet frame transmission is deferred.

4.2 PHY Interrupts

SEI implements a “Link Status Changed” interrupt. Whenever the link status changes, bit LSCIF in IF register is set and if enabled, an external interrupt is issued.

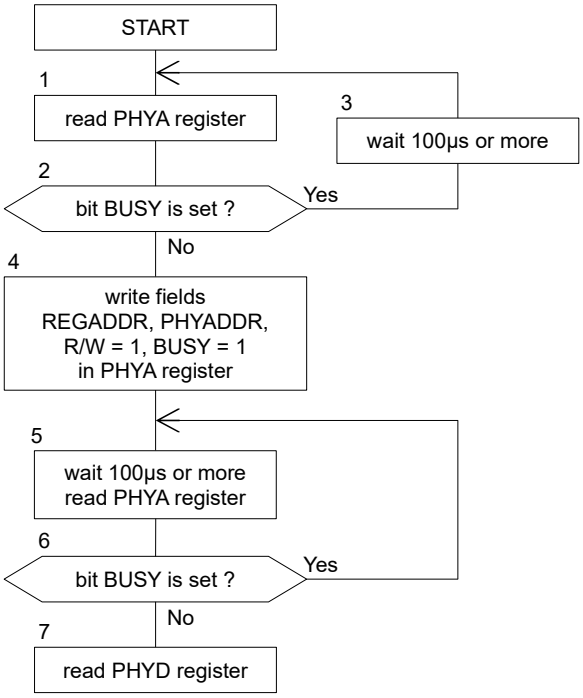
4.3 HC access to PHY

SEI handles only the basic PHY functionality. In order for the HC to access and control more features a PHY can provide, a register based mechanism is implemented so HC can read / write all 32 PHY registers. There are two registers involved: PHYA (PHY Address) and PHYD (PHY Data).

To read a PHY device register follow these steps:

1. read PHYA register
2. if bit BUSY is clear, skip next step
3. wait 100µs or more and then go to 1
4. write register PHY device address to bit field REGADDR, PHY device address to bit field PHYADDR, set bit R/W, BUSY in PHYA register, all at once
5. wait 100µs or more and then read PHYA register
6. if bit BUSY is set, proceed to step 5
7. read register PHYD

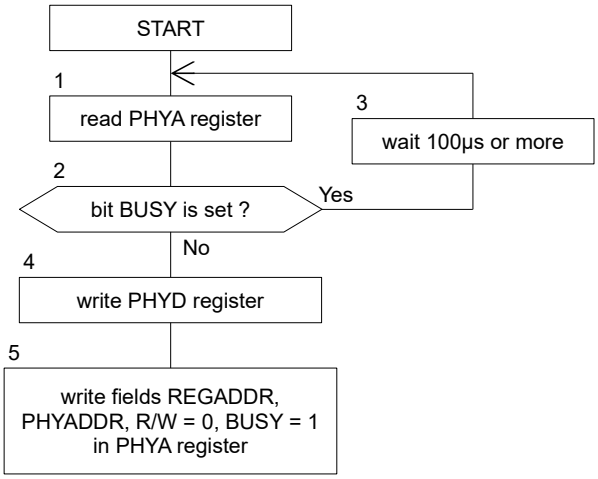
Fig. 4.3-1: Read a PHY device register flowchart



To write a PHY device register follow these steps:

1. read PHYA register
2. if bit BUSY is clear, skip next step
3. wait 100µs or more and then go to 1
4. write register PHYD with the value to be written to the PHY device register
5. write register PHY device address to bit field REGADDR, PHY device address to bit field PHYADDR, write zero to bit R/W, write 1 to bit BUSY in PHYA register, all at once

Fig. 4.3-2: Read a PHY device register flowchart



5 ARP – Address Resolution Protocol

5.1 ARP Overview

The ARP protocol is handled by a routine called here ARP which among other routines gets serviced by the CPU in a round-robin fashion.

An Ethernet frame which is part of the Data Link Layer just on top of the physical layer contains among other, two fields: destination MAC address and source MAC address.

PIC32 microcontrollers come with a preassigned MAC address in the form of three 16 bit registers: EMAC1SA0, EMAC1SA1 and EMAC1SA2. It is this MAC address that SEI uses when sends out an Ethernet frame. HC can read this preassigned MAC address by reading HMA register. HC can also assign a new MAC address by writing to the same HMA register. Note that this new MAC address is not persistent, in other words it is lost when device loses power.

5.2 ARP Reply

On power on, SEI does not have an IP address assigned, therefore it does not respond to ARP Request messages.

SEI sockets can each have a different local IP address. This makes possible for SEI to be part of multiple subnets.

When an ARP Request message arrives, SEI check if HIPA register value is a match. If yes, an ARP Reply message is sent. If there is no match, SEI scans all sockets, regardless if enabled or not, for a match with the SnIPA register. If there is a match, an ARP Reply message is sent. It is important that disabled sockets to have their SnIPA register cleared. For this reason, when HC disables a socket, SEI clears its SnIPA register.

5.3 ARP Request

The interaction between ARP and sockets is made around an ARP Table. When ARP obtains a MAC address, it writes it to a row in the ARP Table. When a socket is about to send an Ethernet packet, it looks in the ARP Table for a row that contains a MAC address and the destination IP address the packet is to be sent.

The ARP Table has a fixed number of rows equal with the number of sockets implemented. A row has three columns: Status 16bits, IP address 32bits and MAC 48bits.

A row in the table can be in one of the following states:

- empty
- timeout
- acquiring
- acquired
- static

When in empty state, the value of the MAC field is not defined.

When in timeout state it means that ARP failed to acquire a MAC address.

When in acquiring state ARP send an ARP Request message and waits for a response. If a response arrives, it changes state to acquired. If a reply does not arrive in time, it changes state to timeout.

When in acquired state, the MAC column has the MAC address that a socket can use. When in this state, ARP runs a “time to live” timer for the respective row. When the timer expires, the row switches to the empty state. Register ARPC (ARP Control) controls the behavior of the timer. The options are that the timer can be reset when a packet is sent, or a packet is received, never reset or never let to expire.

When in static state, the entry never expires. The row enter this state when HC manually writes a MAC address to the SnPMAC register.

The time to live timer is 13bits wide and is incremented by one every second. When a packet is received or sent and ARP is set to reset the timer, only the eight upper bits are cleared. This gives the timer a new somehow random value.

A row in the ARP Table has a weak association with the socket with the same number. When a socket looks for a MAC address in the table, it first looks in the associated row. If not found, it scans all rows from top down. If no match is found, then it places a request on the row withe same number as the socket. While waiting for a MAC address, bit WA (bit 14) in SnCS (Socket n Control and Status register) is set and reset when the MAC is resolved.

When ARP resolves a MAC address, it sends an ARP Request message for a number of times. This is controlled by the field ARR inn ARPC register.

6 ICMP

6.1 ICMP Overview

SEI uses the ICMP protocol to send and receive error and information messages.

6.2 Receiving ICMP messages

SEI handles the following ICMP messages:

– type 3, code any: Destination Unreachable
– type 8, code 0: Echo Request (Ping)

Messages not listed in the above list are ignored.

6.2.1 Receiving ICMP message type 3

When an ICMP message type 3 is received, SEI performs the following:

– if bit ICRIE in ICMP register is set, it sets bit ICIF in IF register and if bit IPINE in GC register is set, it issues an interrupt
– if bit ICRF in ICMP register is not set, it saves up to 56 bytes of the received message in a buffer that HC can read via ICMPR register and then sets bit ICRF. Note that this operation is not atomic. The HC should check bit ICRF first and only if it is set, to read ICMPR register. ICMPR register must be read in full otherwise bit ICRF would not clear.

6.2.2 Receiving ICMP message type 8

When an ICMP type 8 message is received, SEI performs the following:

– it checks if the destination IP address of the received ICMP message matches the value in HIPA register. If it is not a match, the received ICMCP message is ignored. Note that if a Ping Request message is addressed to a socket that has a different source IP address (SnSIPA register) than the Host IP Address (HIPA register), the request message is also ignored. In other words, SEI only replies to ping requests addressed to the IP address in the HIPA register.
– it checks if the message payload data length is between 32 and 64 bytes. If it is not, the received ICMCP message is ignored
– if bit PQRIE in ICMP register is set, it sets bit ICIF in IF register and if bit IPINE in GC register is set, it triggers an interrupt
– if bit PQRF in ICMP register is not set, it updates register PQIP with the source IP address of the received ICMP message (ping request) and then sets bit PQRF. Note that this operation is not atomic. HC should read register PQIP only if bit PQRF is set.
– If bit PRE in ICMP register is set, it replies with an ICMP type 0 (ping reply) message

Note that HC can be notified of a Ping Request message received even if SEI is set not to reply.

6.3 Sending ICMP messages

SEI sends the following types of ICMP messages:

– type 0, code 0: Echo Reply (Ping)
– type 3, code 3: Destination port unreachable

6.3.1 Sending ICMP message Echo Reply

When conditions to send a Ping Reply message are met (see 6.2.2 Receiving ICMP message type 8 above), an ICMP Ping Reply message is written to the Ethernet transmit buffer. However, if at the time the buffer is full or Link is down, the operation is cancelled. ICMP messages to be sent are not buffered. Therefore, in extreme conditions, it is possible for a Ping Reply message to fail. The user should not qualify a network connection performance based on this Ping functionality.

6.3.2 Sending ICMP message type 3, code 3

SEI sends an ICMP message type 3 code 3 in response to a received UDP packet for which there is no enabled socket that matches the destination IP – Port pair. For TCP packets it replies with a TCP - RST packet.

SEI copies the first 28 bytes from the received UDP packet starting with the IP header and then it appends to the ICMP message.

7 DHCP

7.1 DHCP Overview

The user has the option to configure the SEI with a static IP address of to use the DHCP client (called here “DHCP”) for a dynamic configuration. The dynamic configuration requires a DHCP server on the local network.

7.2 Using DHCP

On power on or after a reset, DHCP client is disabled. To enable DHCP client, set bit DHEN in DHCS register. To disable DHCP client, simply write a zero to bit DHEN.

While DHCP acquires configuration settings from a DHCP server, the HC can follow the process by reading the status bit field DHS in DHCS register or using the interrupt system.

DHS = 0 DHCP disabled

While disabled, DHS status reads zero. Once bit DHEN is set, DHCP clears registers HIPA, HSM, GIPA, DHSRV, DNS1 and DNS2 and sets DHS to 1. If SEI was configured before, HC should stop using the previous IP address until the configuration completes.

DHS = 1 Discovery

DHCP client sends a DHCPDISCOVERY message. If a DHCPOFFER message is received, DHS field is set to 2 and proceeds to the next step. If no DHCPOFFER message is received, the DHCPDISCOVERY is resent after a delay. DHCP will keep sending DHCPDISCOVERY messages at a doubled interval each time until it exceeds 60 seconds. If no offer is received, it enters state 6, bit DHCPIF bit in IF register is set and if bit DHIE in DHCS register is set, an interrupt is issued. DHCP will stay in this state until HC disables it.

DHS = 2 Request

DHCP client has received an offer and now sends a DHCPREQUEST message. DHCP requests the following data - beside an IP address which is received by default:

- subnet mask
- gateway IP address
- up to two DNS IP addresses if available
- lease time
- renew time
- rebind time

If it receives an acknowledgement (DHCPACK message), it enters state 3 – Settings Acquired.

If it receives a not acknowledgement (DHCPNAK message), it enters state 6, bit DHCPIF bit in IF register is set and if bit DHIE in DHCS register is set, an interrupt is issued. DHCP will stay in this state until HC disables it.

If no DHCPACK message is received, the DHCPREQUEST message is resent after a delay. DHCP will resend DHCPREQUEST messages at a doubled interval each time until it exceeds 60 seconds. If no DHCPACK message is received, it enters state 6, bit DHCPIF bit in IF register is set and if bit DHIE in DHCS register is set, an interrupt is issued. DHCP will stay in this state until HC disables it.

DHS = 3 Settings Aquired

DHCP has received a DHCPACK message with data requested or part of it. DHCP takes the following steps:

- updates registers HIPA, HSM, GIPA, DHSRV and DHLRM
- updates registers DNS1 and possibly DNS2 if option 6 has been received

- updates renewal time T1 with value from option 58 if received, otherwise it sets to half of lease time
- updates rebinding time T2 with value from option 59 if received, otherwise it sets to 0.875 of lease time
- it sets bit DHCPIF in IF register and if bit DHIE in DHCS register is set, it issues an interrupt
- sets DHS bit field to 3

DHCP stays in this state until the renewal time T1 expires when it enters Renewal state of HC disables DHCP.

From now on SEI is configured with a dynamic IP address and HC can start using it. Most of the time DHCP stays in this state and this is when DHCP interrupt fires if enabled.

When T1 expires, it enters the Renewal state.

DHS = 4 Renewal

DHCP sends a unicast DHCPREQUEST message. The message is sent to the same DHCP server that granted the IP address.

If a DHCPACK message is received, it updates the timers and goes back to state 3 – Settings Acquired. This is the most likely scenario. HC is not notified about this step.

If it receives a not acknowledgement (DHCPNAK message), it enters state 6, bit DHCPIF bit in IF register is set and if bit DHIE in DHCS register is set, an interrupt is issued. DHCP will stay in this state until HC disables it.

If no DHCPACK message is received, the DHCPREQUEST message is resent after a delay. DHCP will resend DHCPREQUEST messages at a doubled interval each time with an upper limit of 60 seconds until T2 expires. If T2 expires, it enters the Rebinding state.

DHS = 5 Rebinding

DHCP sends a broadcast DHCPREQUEST message. If it receives a DHCPACK message it updates the timers and transitions to state 3 - Settings Acquired. HC is not notified about this event.

If it receives a not acknowledgement (DHCPNAK message), it enters state 6, bit DHCPIF bit in IF register is set and if bit DHIE in DHCS register is set, an interrupt is issued. DHCP will stay in this state until HC disables it.

If no DHCPACK message is received, the DHCPREQUEST message is resent after a delay. DHCP will resend DHCPREQUEST messages at a doubled interval each time with an upper limit of 60 seconds until the lease time expires. If the lease time expires, it enters state 6, bit DHCPIF bit in IF register is set and if bit DHIE in DHCS register is set, an interrupt is issued. DHCP will stay in this state until HC disables it.

DHS = 6 or 7 Timeout, Error

DHCP enters state 6 or 7 due to a timeout or an error with the DHCP protocol. In this state HC should not use a previously acquired dynamic IP address. DHCP will remain in this state until HC takes action. The only action that HC can take is to disable DHCP and then optionally to enable it back.

8 GPIO

8.1 GPIO Overview

SEI provides a mechanism for HC to handle 15 bits on port B and 8 bits on port E. Port B is handled like any other 16 bit register. Port E is a special case where it is handled by the only 16bit SPI transaction. On start up, register AD1PCFG is set to 0xFFFF, meaning that port B inputs are set to digital mode.

The default configuration for these two ports is digital input. Pins unconnected should not be left floating. They can be configured as outputs of tied to GND of VCC with a resistor.

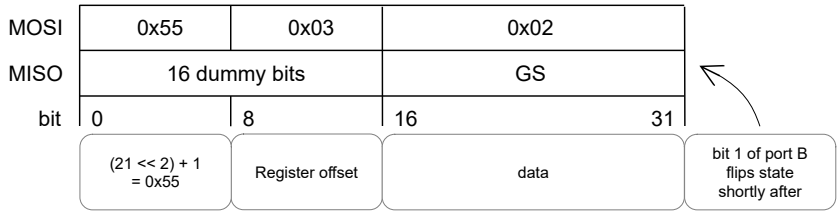
There are four registers implemented for each port:

– TRISB, TRISE set digital direction to input or output
– LATB and LATE control the output pin state
– PORTB, PORTE read the pin state
– ODCB, ODCE set digital output pin configuration to push-pull or open drain.

There are three associated registers CLR, SET and INV with each register described above, except for PORTB and PORTE which are read only. These associated registers are accessible using an offset written to the second byte of the SPI header.

Example: flip bit 1 on port B: SPI command code for register LATB is 21. LATBINV associated register is at an offset of 3.

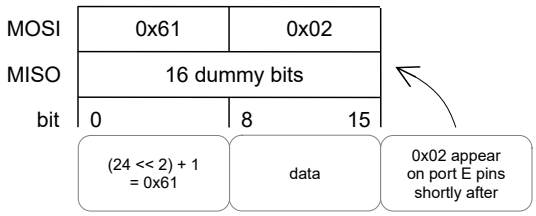
Fig. 8.1-1: flip bit 1 on port B



Port E registers, except PORTE, are written to the SPI header second byte location. This is the only SPI transaction 16 bit long. HC can continue to write new values as pairs of 16 bits during the same SPI transaction. SEI updates the respective register with the upper byte, and after 100ns with the low byte. Reading a port E register is the same as any 16 bit register read. The actual value is on the lower byte while the high byte reads always zero. There are no CLR, SET or INV options for port E registers.

Example: set port E output to 0x02: SPI command code for register LATE is 24.

Fig. 8.1-1: Set port E output to 0x02



9 Common registers detailed description

Legend:
- R – readable bit
- W – writable bit
- U – unimplemented
- n – value at POR
- u – value unknown
- 0 – bit cleared
- 1 – bit set

GC

General Control register. Command code: 0. Size: 2 bytes.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	R/W-0	U-0	R/W-1	U-0	U-0	U-0	U-0	U-0
	SWRST	SPIM	PHRST	-	-	-	-	-
7÷0	U-0	U-0	U-0	R/W-0	U-0	U-0	U-0	R/W-1
	-	-	-	IPINE	-	-	-	RMIEN

bit 15 **SWRST:** To soft-reset SEI, write 0x8000 to this register and 0xAAAD to SMR register in one SPI transaction. The value of this register does not change following this operation.

bit 14 **SPIM:** To change the SPI mode, write 0x4000 to this register and the desired value to SMR register in one SPI transaction. See table 8-1 for details. The value of this register does not change following this operation.

bit 13 **PHRST:** Phy Reset. To change the state of pin PHRST, write the corresponding value to SMR register in one SPI transaction. See table 8-1 for options. The value of this register does not change following this operation. Note that some PHY devices when under reset conditions no longer provide the 50MHz clock. If SEI is using this clock, a lock-up state can occur.

bit 12÷5 These bits are not used. For future compatibility write 0.

bit 4 **IPINE:** Interrupt Pin Enable
1: Interrupt pin enable
0: Interrupt pin disable

bit 3÷2 These bits are not used. For future compatibility write 0.

bit 0 **RMIEN:** Register Monitor Interface Enable

To soft-reset the device, bit RESET must be the only one set. When changing the SPI mode, bit SIPM must be the only one set. When handling the PHRST pin, bit PHRST must be the only one set. In this case the other bits in the register are not affected. When bit SWRST, SPIM or PHRST are set, HC must continue and write to register SMR.

GS

General Status register. This register is read only. It is read with every SPI transaction. There is no particular SPI command that specifically reads this register. Size: 2 bytes.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	R-x	R-x	R-x	R-x	R-x	U-0	R-1	R-1
	EXTR	SWR	WDTO	BOR	POR	-	CPOL	CPHA
7÷0	U-0	R-0	U-0	R-1	U-0	U-0	U-0	U-0
	-	-	LKUP	IPINS	-	-	-	-

bit 15÷10 These bits are not used.

bit 9 **SCPOL:** SPI Clock Polarity. This bit is read only. To change the SPI mode, see register SMR.

bit 8 **SPHA:** SPI Clock Phase. This bit is read only. To change the SPI mode, see register SMR.

bit 7÷6 These bits are not used.

bit 5 **LKUP:** Link UP.
0: Link is down
1: Link is up

bit 4 **IPINS:** Interrupt Pin Status.
0: Interrupt Pin is low
1: Interrupt Pin is high (with the help of a pull-up resistor)

bit 3÷0 These bits are not used.

SMR

SPI Mode and Reset register. Command code: 0. Size: 2 bytes. This register is write only. It is used to soft reset SEI, change the SPI mode or to handle the PHY reset pin. When HC resets SEI by writing to this register, SPI mode switches to mode 3.

The following table describes the values to be written. SEI ignores all other values.

Table 8-1: Register SMR values

Task	SMR
SWRST	0xAAAD
SPIM0	0xAAA8
SPIM1	0xAAA9
SPIM2	0xAAAA
SPIM3	0xAAAB
PHRST clear	0xAAA0
PHRST set	0xAAA1

HMAC

Hardware MAC address register. Command code: 1. Size: 6 bytes. This register holds the MAC address for this device.

Right after reset, this register is loaded with the hard-coded MAC address listed under the Microchip Technology name. The host controller can write a different MAC address to this register. This register can be reloaded with the hard-coded one by writing 00:00:00:00:00:00 to it. The host controller should update this register only when all sockets are disabled.

R/W	R/W	R/W	R/W	R/W	R/W
byte 1	byte 2	byte 3	byte 4	byte 5	byte 6

HIPA

Host IP Address register. Command code: 2. Size: 4 bytes. This register holds the Host IP address and it is used by ARP, Ping algorithm and DHCP. Sockets have their own local IP address register.

R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4

HSM

Host Subnet Mask register. Command code: 2. Size: 4 bytes. This register holds the host subnet mask.

R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4

GIPA

Gateway IP address. Command code: 2. Size: 4 bytes. This register holds the gateway IP address.

R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4

EPR

Ephemeral Port Range. Command code: 3. Size: 4 bytes. This register holds two 16bit registers: EPS and EPE. EPS represents the lower limit for the ephemeral port range while EPE represents the upper limit. By default this range goes from 49152 to 65535. Both 16 bit registers are written after all 32 bit data is clocked in.

When writing to this register, EPS and EPE must satisfy the next conditions:

1: 1025 <= EPS <= 49152

2: 5000 <= EPE <= 65535

3: EPE – EPS >= 3976.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
31÷24	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
	EPE1<7:0>							
23÷16	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
	EPE0<7:0>							
15÷8	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	EPS1<7:0>							
7÷0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	EPS0<7:0>							

bit 31÷24 **EPE1<7:0>:** EPE byte 1

bit 23÷16 **EPE0<7:0>:** EPE byte 0

bit 15÷8 **EPS1<7:0>:** EPS byte 1

bit 7÷0 **EPS0<7:0>:** EPS byte 0

IF

Interrupt Flags register. Command code: 4. Size: 4 bytes. This register is read only. This register is cleared after the register is read. Reading this register also puts the external interrupt line in logic high.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
3	S15IF	S14IF	S13IF	S12IF	S11IF	S10IF	S9IF	S8IF
2	S7IF	S6IF	S5IF	S4IF	S3IF	S2IF	S1IF	S0IF
1	-	-	-	-	-	-	-	-
0	-	-	LSCIF	-	-	ATOIF	ICIF	DHIF

bit 31÷16 **SnIF:** Socket n Interrupt flag

bit 15÷6 Unimplemented. Read as 0.

bit 5 **LSCIF:** Link Status Changed Interrupt Flag.

bit 4÷3 Unimplemented. Read as 0.

bit 2 **ATOIF:** ARP Timeout Interrupt Flag.

bit 1 **ICIF:** ICMP Interrupt Flag.

bit 0 **DHIF:** DHCP Status Changed Interrupt Flag.

PHYCS

PHY Control and Status register. Command code: 5. Size: 2 bytes. Control bits are located in the low byte while the status bits are located in the high byte.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	U-0	U-0	R-0	U-0	R-0	U-0	U-0	R-0
	-	-	LSPS	-	PHPDS	-	-	DPLXS
7÷0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-1	R/W-1
	BUSY	RW	LSCIE	AN	PHPD	PHSR	LSP	DPLX

bit 15÷14 These bits are not used. For future compatibility write 0

bit 13 **LSPS:** Link Speed Status. This bit value is relevant only when link is up (bit LKUP in GS register is set).
0: 10Mbps
1: 100Mbps

bit 12 This bit is not used. For future compatibility write 0.

bit 11 **PHPDS:** PHY Power Down Status.
0: PHY is not in Power Down mode
1: PHY is in Power Down mode.

bit 10÷9 These bits are not used. For future compatibility write 0

bit 8 **DPLXS:** Duplex Status. This bit value is relevant only when bit LKUP in GS register is set.
0: Half duplex
1: Full duplex

bit 7 **BUSY:** Read / Write Busy. Set this bit in order to start a read / write operation. This bit clears itself once the operation completes. Writes to this register while this bit is already set are ignored.

bit 6 **Read/Write:** Set this bit if a read is to be performed. Clear this bit if a write is to be performed.
0: write
1: read.

bit 5 **LSCIE:** Link Status Changed Interrupt Enable.
0: Disable Link Status Changed Interrupt
1: Enable Link Status Changed Interrupt

bit 4 **AN:** Auto Negotiation:
0: Turn off auto negotiation
1: Turn on auto negotiation

bit 3 **PHPD:** PHY Power Down. Set this bit in order to put the PHY interface in Power Down mode. Note that the Power Down characteristics / behaviour varies among devices. Perform a custom PHY register read / write in order to explore more options (if any) on a particular PHY device.
0: Normal operation
1: Power Down.

bit 2 **PHSR:** PHY Software Reset. Set this bit in order to soft-reset the PHY device. This bit can only be set. This bit clears itself once the operation is complete.
0: No action taken
1: Reset PHY device.

bit 1 **LSP:** Link Speed. Set this bit so the PHY interface will run at 100Mbps. Clear this bit to limit the speed on the wire to 10Mbps. This bit is relevant only when bit AN is not set.

bit 0 **DPLX:** Duplex mode. This bit is relevant only when bit AN is not set.
0: Half Duplex.
1: Full Duplex.

PHYA

PHY Address register. Command code: 6. Size: 2 bytes.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	-	-	-	PHYADDR				
7÷0	R/W1-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	-	-	-	REGADDR				

bit 15÷13 These bits are not used. For future compatibility write 0.

bit 12÷8 **PHYADDR:** PHY Address register. Write these bits with the PHY address to be read / written.

bit 7÷5 These bits are not used. For future compatibility write 0.

bit 4÷0 **REGADDR:** PHY Register Address. Write these bits with the PHY register's address to be read / written.

PHYD

PHY Data register. Command code: 7. Size: 2 bytes. Writes to this register while bit BUSY in PHYC register is set are ignored. When bit BUSY in PHYC register is set, the value of this register is not valid.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	PHYD1<7:0>							
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	PHYD0<7:0>							

bit 15÷8 **PHYD1:** PHYD byte 1. For implementations with a network switch as KSZ8863RLL, this byte is not used.

bit 7÷0 **PHYD0:** PHYD byte 0.

ICMP

ICMP Control and Status register. Command code: 8. Size: 2 bytes.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-1	R/W-0
	-	-	-	-	-	ICSE	PRE	ICIE
0	U-0	U-0	U-0	U-0	U-0	R-0	R-0	R-0
	-	-	-	-	-	ICSF	ICRF	PQRF

bit 15÷11 These bits are not used. For future compatibility write 0.

bit 10 **ICSE:** ICMP packet Send Enable. This bit enables sending ICMP messages. HC should keep this bit set.

bit 9 **PRE:** Ping Reply Enable. If this bit is set, SEI replies to PING Query messages.

bit 8 **ICIE:** ICMP Interrupt Enable.

bit 7÷3 These bits are not used. For future compatibility write 0.

bit 2 **ICSF:** ICMP packet Sent. This bit is set when an ICMP message is to be sent. This bit clears itself when register ICPS is read.

bit 1 **ICRF:** ICMP packet Received. This bit is set when an ICMP message type 3 is received. This bit clears itself when register ICPR is read.

bit 0 **PQRF:** PING Query Received Flag. This bit is set when an ICMP Ping Query message is received. This bit clears itself when register PQIP is read.

PQIP

Ping Query IP address register. Command code: 9. Size: 4 bytes. Read this register in order to find out the originating IP address of the last Ping Query message received. This register is cleared by reading it. Reading a value of 0:0:0:0 denotes that no Ping Query message has been received.

R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4

ICMPS

ICMP packet Sent. Command code: 10. Size: variable. This register holds the last ICMP message sent. The first two bytes holds the length of data that follows. These first two bytes are cleared after reading this register. Therefore if the first two bytes comes out as zero, it denotes that the remaining data is not valid.

ICMPR

ICMP packet Received. Command code: 11. Size: variable. This register holds the last ICMP message received (except for Ping Request messages). The first two bytes holds the length of data that follows. These first two bytes are cleared after reading this register. Therefore if the first two bytes come out as zero, it denotes that the remaining data is not valid.

DHCS

DHCP Control and Status register. Command code: 12. Size: 2 bytes.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15:8	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
	DHEN	-	-	-	-	-	-	DHIE
7:0	U-0	U-0	U-0	U-0	U-0	R-0	R-0	R-0
	-	-	-	-	-	DHS		

bit 15 **DHEN:** DHCP Enable. Set this bit in order to enable the DHCP client.

bit 14÷9 These bits are not used. For future compatibility write 0.

bit 8 **DHIE:** DHCP Interrupt Enable. Needed ?

bit 7÷3 These bits are not used. For future compatibility write 0.

Bit 2÷0 **DHS:** DHCP Status:
0: disabled
1: discovery
2: request
3: settings acquired
4: renew
5: rebind
6: DHCP server error
7: network error

DHSRV

DhcpServer IP addresses. Command code: 13. Size: 4 bytes. DHSRV holds the IP address of DHCP server that responded to DHCPDISCOVERY message. DHCP uses this register value for Renewal and Rebinding process.

R-0	R-0	R-0	R-0
byte 1	byte 2	byte 3	byte 4

DNS1, DNS2

DNS1 and DNS2 IP addresses. Command code: 13. Size: 4 bytes each. DNS1 and DNS2 are returned by the DHCP server with option 6. A DHCP server does not replies with an option 6 if does not have this information. If DNS1 or DNS2 reads 0.0.0.0, denotes that the value has not been provided. SEI does not use these registers other than having them available for HC. HC can use them to implement a DNS protocol.

R-0	R-0	R-0	R-0
byte 1	byte 2	byte 3	byte 4

DHRLT

DHCP Remaining Lease Time. Command code: 14. Size: 4 bytes. DHRLT represents the remaining lease time in minutes. Providing that the Renewal and Rebinding process will fail, the lease time expires after this number of minutes. This register is followed by register DHLT during the same SPI transaction.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
3	R-0							
	DHRLT3<31:24>							
2	R-0							
	DHRLT2<23:16>							
1	R-0							
	DHRLT1<15:8>							
0	R-0							
	DHRLT0<7:0>							

bit 31÷24 **DHRLT3:** DHRLT byte 3.

bit 23÷16 **DHRLT2:** DHRLT byte 2.

bit 15÷8 **DHRLT1:** DHRLT byte 1.

bit 7÷0 **DHRLT0:** DHRLT byte 0.

DHLT

DHCP Lease Time. Command code: 14. Size: 4 bytes. DHLT represents the lease time in minutes obtained from the DHCP server. When read, this register follows register DHRLT.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
3	R-0							
	DHLT3<31÷24>							
2	R-0							
	DHLT2<23÷16>							
1	R-0							
	DHLT1<15÷8>							
0	R-0							
	DHLT0<7÷0>							

bit 31÷24 **DHLT3:** DHLT byte 3.

bit 23÷16 **DHLT2:** DHLT byte 2.

bit 15÷8 **DHLT1:** DHLT byte 1.

bit 7÷0 **DHLT0:** DHLT byte 0.

ARPC

ARP Control register. Command code: 15. Size: 2 bytes.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	R/W-1	U-0	U-0	U-0	R/W-0	R/W-0	R/W-1	R/W-1
	AEN	-	-	-	ARR			
0	R/W-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-1
	CAT	-	-	-	-	ASR	CTTX	CTRX

bit 15 **AEN:** ARP Enable. Write 0 to disable ARP. Write 1 to enable ARP.

bit 14÷12 These bits are not used. For future compatibility write zero.

bit 11÷8 **ARR:** ARP Retransmit. This bit field defines how many times ARP retransmits a request. Writing zero to this bit field is seen as one.

bit 7 **CAT:** Clear ARP Table. Write 1 to clear the entire ARP table. Writing a zero to this bit has no effect. This bit clears itself once the operation is complete. Bit AEN (bit 15) must be set in order for this operation to succeed.

bit 6÷3 These bits are not used. For future compatibility write zero.

bit 2 **ASR:** ARP Static Row. When this bit is set, the rows in the table are static, or in other words they do not expire.

bit 1 **CTTX:** Clear Timer on Transmit. When this bit is set, ARP clears the upper 8 bits of 13 bits time-to-live counter every time a packet is sent.

bit 0 **CTRX:** Clear Timer on Receive. When this bit is set, ARP clears the upper 8 bits of 13 bits time-to-live counter every time a packet is received.

ATTL

ARP Time-To-Live register. Command code: 16. Size: 2 bytes.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	-	-	-	ATTL<14÷8>				
0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0
	ATTL<7÷0>							

bit 15÷13 These bits are not used. For future compatibility write zero.

bit 12÷0 **ATTL:** ARP Time-To-Live. When a MAC address is resolved, a timer that gets incremented every second is started. When the timer reaches the value in this bit field, the row in the ARP table switches status to “empty” and the IP field set to 0.0.0.0 . SEI limits the value that can be written to this bit field to minimum 120 and maximum 8191.

TRISB

Command code: 20. Size: 2 bytes. This register handles the register TRISB in the microcontroller. Only the first 15 bits are available. SEI ignores writes to bit 15 and reads as zero. All three associated registers TRISBCLR, TRISBSET and TRISBINV can be accessed by writing an offset in the second byte of SPI command header. Register TRISBCLR is at offset 1, TRISBSET at offset 2 and TRISBINV at offset 3.

When reading this register, SEI ignores the offset which means that only register TRISB can be read.

Once the SPI header is clocked in, HC can write this register multiple times during the same SPI transaction.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	U-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0
	-	TRISB<14÷8>						
0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0
	TRISB<7÷0>							

bit 15 This bit is not used. For future compatibility write zero.

bit 15÷8 **TRISB 14÷8:** Write 1 to these bits to configure the pins as inputs or 0 as outputs

bit 7÷0 **TRISB 7÷0:** Write 1 to these bits to configure the pins as inputs or 0 as outputs

LATB

Command code: 21. Size: 2 bytes. This register is write only and handles the register LATB in the microcontroller. Only the first 15 bits are available. SEI ignores writes to bit 15. All three associated registers LATBCLR, LATBSET and LATBINV can be accessed by writing an offset in the second byte of SPI command header. Register LATBCLR is at offset 1, LATBSET at offset 2 and LATBINV at offset 3.

Once the SPI header is clocked in, HC can write this register multiple times during the same SPI transaction.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	U-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
	-	LATB<14÷8>						
0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
	LATB<7÷0>							

bit 15 This bit is not used. For future compatibility write zero.

bit 15÷8 **LATB 14÷8:** Write 0 to these bits to set the corresponding pins to logic low or one for logic high

bit 7÷0 **LATB7÷0:** Write 0 to these bits to set the corresponding pins to logic low or one for logic high

PORTB

Command code: 21. Size: 2 bytes. This register is read only and handles the register PORTB in the microcontroller. Only the first 15 bits are available. Bit 15 reads always zero.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	U-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
	-	PORTB<14÷8>						
0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
	PORTB<7÷0>							

bit 15 This bit is not used. For future compatibility write zero.

bit 15÷8 **PORTB 14÷8:** These bits represent the logic state of pins 15÷8.

bit 7÷0 **PORTB7÷0:** These bits represent the logic state of pins 7÷0.

ODCB

Command code: 22. Size: 2 bytes. This register handles the register ODCB in the microcontroller. Only the first 15 bits are available. SEI ignores writes to bit 15 and reads as zero. All three associated registers ODCBCLR, ODCBSET and ODCBINV can be accessed by writing an offset in the second byte of SPI command header. Register ODCBCLR is at offset 1, ODCBSET at offset 2 and ODCBINV at offset 3.

When reading this register, SEI ignores the offset which means that only register ODCB can be read.

Once the SPI header is clocked in, HC can write this register multiple times during the same SPI transaction.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	U-0	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1
	-	ODCB<14÷8>						
0	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1
	ODCB<7÷0>							

bit 15 This bit is not used. For future compatibility write zero.

bit 15÷8 **ODCB 14÷8:** Write 1 to these bits to configure the pins as open drain

bit 7÷0 **ODCB 7÷0:** Write 1 to these bits to configure the pins as open drain

TRISE

Command code: 23. Size: 1 byte. HC writes to this register placing the eight bit value in the second byte of the SPI header transaction. Once all 16 bits of SPI header are in, HC can continue to write to this register in pairs of two bytes.

Reading this register is done as a 16 bit register read, with the actual value in the lower byte.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1
	TRISE<7÷0>							

bit 7÷0 **TRISE 7÷0:** Write 1 to these bits to configure the pins as inputs or 0 as outputs

LATE

Command code: 24. Size: 1 byte. This register is write only. HC writes to this register placing the eight bit value in the second byte of the SPI header transaction. Once all 16 bits of SPI header are in, HC can continue to write to this register in pairs of two bytes.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1
	LATE<7÷0>							

bit 7÷0 **LATE 7÷0:** Write 0 to these bits to set the corresponding pins to logic low or one for logic high

PORTE

Command code: 24. Size: 1 byte. This register is read only and handles the register PORTE in the microcontroller. Reading this register is done as a 16 bit register read, with the actual value in the lower byte.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
	PORTE<7÷0>							

bit 7÷0 **PORTE7÷0:** These bits represent the logic state of pins 7÷0.

ODCE

Command code: 25. Size: 1 byte. HC writes to this register placing the eight bit value in the second byte of the SPI header transaction. Once all 16 bits of SPI header are in, HC can continue to write to this register in pairs of two bytes.
Reading this register is done as a 16 bit register read, with the actual value in the lower byte.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1	RW-1
	ODCE<7÷0>							

bit 7÷0 **ODCE 7÷0:** Write 1 to these bits to configure the pins as open drain

FWIDB

Firmware ID and Build. Command code: 63. Size: 4 bytes. This register holds the firmware ID and the version (build).

byte offset	byte 3/1	byte 2/0
2	R	R
	FWID1	FWID0
0	R	R
	B1	B0

bit 31÷24 **FWID1:** Firmware ID byte 1

bit 23÷16 **FWID0:** Firmware ID byte 0

bit 15÷8 **B1:** Build byte 1

bit 7÷0 **B0:** Build byte 0

10 UDP – User Data Protocol

10.1 UDP Overview

UDP and TCP are two of the most used data transport protocols. Two main things differentiate them:

One is that in case of UDP data arrives packed in the same way it was sent, or in other words data is attached to the particular packet. This simplifies reading data because specific data is at the same offset it was sent. TCP is based on a data stream, therefore packets can arrive in different sizes so the beginning of a packet has no meaning.

Another major difference is that UDP does not check if the packet reached the receiver while TCP does, and in case it did not, it retransmits the lost packet.

SEI provides two UDP working modes: client and server. In client mode, SEI exchanges data with only one receiver called here “peer”. Usually the UDP client initiates data transfer but not necessary. In UDP server, SEI accepts exchanging data with multiple peers.

10.2 UDP Client

To exchange data over UDP, HC picks a socket number, configure it by writing to various registers and enable it. Once a socket is enabled, HC can simply write data to be sent to the transmit register. When done, disable the socket.

10.2.1 Enable socket in UDP Client mode

To setup a socket in UDP Client mode, follow the next steps:

1. Make sure register SnCS value is zero. If not, write 0xFFFF to SnCSCLR register and wait for SnCS to be cleared
2. write the source IP address to SnIPA register; usually this is the same with the host IP address in register HIPA
3. write the subnet mask to SnSM register
4. write the peer IP address to SnPIPA register
5. if a specific local port number is preferred, write it to SnP register; if not, leave it as zero and an ephemeral port will be assigned before the first data is sent
6. write the peer port number to SnPP register
7. assign a receive and transmit buffer by writing to SnBSPS; make sure these buffers does not overlap with other enabled socket buffers
8. optionally, HC can write a particular TTL (time-to-live) value to the SnTTL register
9. optionally HC can write a particular value to SnMTU (Maximum Transmission Unit) register. The receiving MTU is always 1500 bytes
10. enable the socket by writing a one to bit SEN (SocketEnable) in register SnCSSET

The Socket Control and Status register SnCS can only be set and clear by writing to SnCSET and SnCCLR. The transfer is made when CPU services the socket routine, therefore is not immediate. HC should wait for this to happen and check for socket status later. The time required can vary from microseconds up to one millisecond; it depends how busy CPU is servicing other tasks. It is fair to poll the status register every 100µs.

Upon enabling the socket SEI takes the following steps:

1. checks if SnPPN (Peer Port Number) register is not zero. If it is, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next steps
2. checks if source and destination IP addresses are in the same subnet

- if yes, the destination IP address is the one that ARP will be required to get a MAC address for

- if not, it checks if the gateway IP address is in the same subnet; if yes, the gateway IP address is the one ARP will be asked to get a MAC address for; if not, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next steps

3. checks if assigned received and transmit buffers are in range of total available memory. If they are in range, proceeds to next step; if not, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next step. SEI does not check if the assigned buffers overlap with other assigned buffers. HC must insure the buffers are assigned right
4. initialize the head and tail receive and transmit pointers and updates the SnCS register as follows:

   - sets bit field SS (SocketStatus) to one, indicating that the socket is running

   - sets SEN bit indicating that the socket is enabled

   - clears bit field MODE indicating that the socket is in UDP Client mode
5. it assigns a row in the ARP table. The row number assigned is the same as the socket number.

From now on HC can send data. Once the destination MAC address and local port number are resolved, HC can receive data as well.

10.2.2 Sending data in UDP Client mode

Data to be sent in a UDP packet must be written to the transmit buffer in one SPI transaction. For this, HC uses command code 13. This command uses a four byte header followed by payload data. If data length is an odd number, it must be padded with a dummy byte to an even number.

Table 10.2.2-1: Command code 13 header structure when writing data to the transmit buffer

Bit offset	Size [bits]	Name	Description
0	6	CC	Command Code: 13
6	1	RW	Read / Write: 0
7	1	RT	Register Type: 0 for socket registers
8	8	SN	Socket Number
16	16	PDL	Payload Data Length

Before writing data to the transmit buffer, HC have to make sure there is enough space in the buffer. There are two ways to do this.

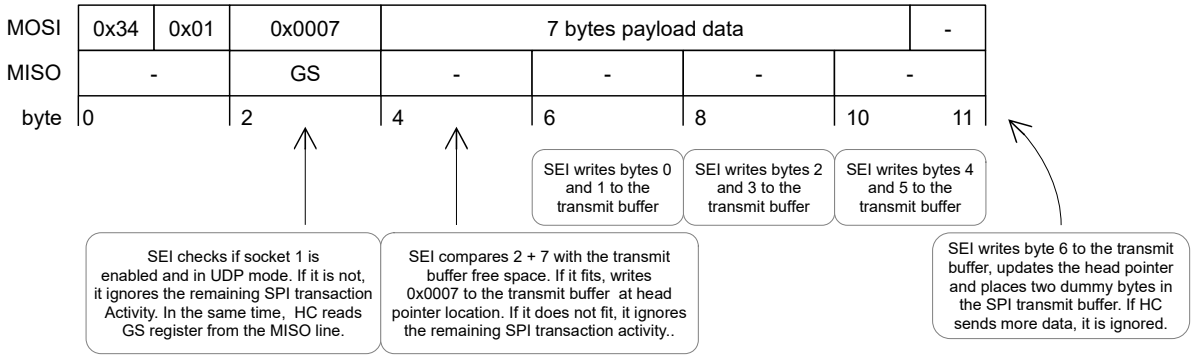
One option is to read the SnTFS (Socket Transmit Buffer Free Size) register and either send less data that would fit or wait until more space is freed. This option fits best when the transmit buffer is large and waiting for it to empty would be inefficient in terms of speed.

The other option is to simply read the SnCS register and look for bit TBE (Transmit Buffer Empty). When this bit is set, the entire transmit buffer is available. This option fits best when the transmit buffer is small. Given the fact that the SPI speed is considerably lower than the one for Ethernet, the transmit buffer is empty most of the time.

If however HC attempts to send more data that the transmit buffer can hold, data is discarded and the head pointer is not advanced. SEI writes a two byte descriptor in the transmit buffer just before data. This descriptor holds the data length that follows. Therefore HC should take this in consideration when evaluating the available space in the buffer.

Data is written to the transmit buffer as it arrives over the SPI bus. Once all data is in, the head pointer is advanced and bit TBE reads zero indicating that the transmit buffer is not empty and data is sent out once the socket gets serviced by CPU. If more data is sent than indicated in the command header, the extra data is ignored.

Fig. 10.2.2-1: Example: Write 7 bytes of data to socket 1 transmit buffer in UDP Client mode



Before first data to be sent over Ethernet, SEI checks the local port number in SnP register. If it is already defined by HC, it proceeds to the next step. If the value in SnP is zero, an ephemeral port is assigned.

Next, SEI looks in the ARP table, specifically if the IP address field in the associated row is the same with the socket destination IP address if in the same subnet, or the Gateway IP address if not in the same subnet. If found, it checks the status filed. If the row contains a valid MAC address, it proceeds to send the packet. If not, it scans the entire ARP table. If a valid row is found, it marks as the associated row, and proceeds to send the packet. If no valid row is found, then an ARP request is placed in the default associated row, and bit WA (Wait ofr ARP) in SnCS register is set.

If the ARP succeeds, the packet is sent and bit in SnCS register cleared. If ARP times out, socket status changes to “ARP Timeout” and the corresponding bit in IF register is set. If the interrupt pin is enabled, an interrupt is issued. In this state, HC can only disable the socket.

Once the packet is sent, SEI advances the tail pointer.

10.2.3 Receiving data in UDP Client mode

For a UDP packet to be received, the following conditions must be true:

- packet destination IP address equals SnIPA register (local IP address)
- packet source IP address equals SnPIPA register (peer IP address)
- packet destination port number equals SnP register (local port number)
- packet source port number equals SnPP register (peer port number)
- there is enough space in the receive buffer to store the payload data

If the conditions above are met, SEI inserts two byte packet descriptor before payload data. The descriptor indicates the data length that follows. SEI then copies the payload data from the Ethernet buffer to the socket receive buffer and advances the head pointer. SEI also sets bit SnIF in IF register and if the interrupt pin is enabled, it issues an interrupt.

The receive buffer is limited and since SPI bus speed is lower than the one on Ethernet side, there is a possibility that the received data cannot be saved. In this case bit PDPIF (Packet Dropped / Push Interrupt Flag) is set, and the corresponding socket interrupt flag in IF register set. If the interrupt pin is enabled, an interrupt is issued.

Beside the interrupt system, there other ways for HC to figure out that there is data to be read in the receive buffer.

HC can poll bit DRB (Data in Receive Buffer) in SnCS register. If it is set, it means that there is at least one packet in the receive buffer.

HC can poll register SnRDS (Socket Received Data Size).

Or HC can simply attempt to read data. If there is no data, SEI indicates this by sending 0xFFFF as data length. The maximum packet data length in UDP mode is 1472 bytes.

HC reads data using command code 13. A whole received data packet must be read at once. This command uses a two byte header. SEI first places two dummy bytes on the MISO line followed by the GC register and then a 16 bit value that indicates the length of the following payload data. Finally data follows. A UDP packet without payload data is a valid one, therefore it is normal that in such cases SEI indicates a zero length. If data length is an odd number, HC must clock in an extra byte to make the total even.

Once the entire packet data is read, SEI advances the tail pointer. If HC keeps reading more data, SEI ignores and data read is not valid.

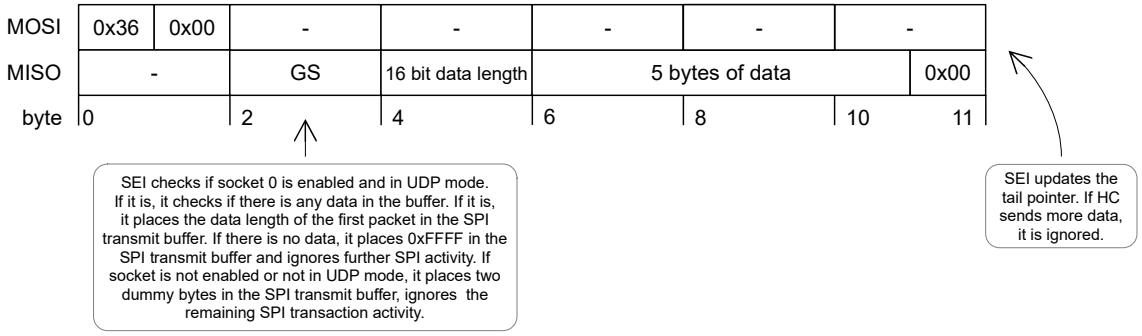
HC should keep reading more data packets until empties the receive buffer in order to avoid overflow.

Bit DRB in SnCS register clears itself when the receive buffer gets empty.

Table 10.2.3-1: Command code 13 header structure when reading data from the receive buffer (UDP Client mode)

Bit offset	Size [bits]	Name	Description
0	6	CC	Command Code: 13
6	1	RW	Read / Write: 1
7	1	RT	Register Type: 0 for socket registers
8	8	SN	Socket Number

Fig. 10.2.3-1: Example: Read a received UDP packet of 5 bytes from socket 0 in UDP Client mode



10.2.4 Disable socket in UDP Client mode

HC can disable the socket by simply writing 0xFFFF to the SnCCLR register. This operation is handled by the socket routine therefore if needed, HC should check for its completion.

When SEI disables the soket, if performs the followings:

– clears the first 12 bits in SnCS register

– clears register SnP (local Port number) suggesting an ephemeral port number next time the socket is enabled in UDP Client mode

A socket requires the least CPU intervention when is disabled, therefore HC should keep unused sockets disabled.

10.3 UDP server

In UPD server HC can exchange data with more than one peer. Usually HC receive data from a peer and replies to it. Because of that, normally HC does not set a destination IP address nor a per port number. It simply replies to the same peer that the message is coming from.

10.3.1 Enable socket in UDP Server mode

To setup a socket in UDP Server mode, follow the next steps:

1. write the source IP address to SnIPA register; usually this is the same with the host IP address in register HIPA
2. write the subnet mask to SnSM register
3. write a local port number to SnP register
4. assign a receive and transmit buffer by writing to SnBSPS; make sure these buffers does not overlap with other enabled socket buffers
5. optionally, HC can write the SnTTL with a TTL (Time To Live) desired value
6. enable the socket by writing the following fields in SnCSET:
   - write 1 to SEN (SocketEnable) bit
   - write 1 to bit MB if peer MAC address is to be included when writing and reading data over the SPI interface
   - write 1 to bit RBA if receiving packets from the broadcast FF.FF.FF.FF IP address is allowed
   - write 1 to bit RSBA if receiving packets from the subnet broadcast IP address is allowed
   - write 1 to MODE field (UDP Server)

   Note that writing to SnCSET register, bits can only be set. To get a fresh SnCS register write 0xFFFF to SnCCLR register before this step

The Socket Control and Status register SnCS can only be written by writing to SnCSET and SnCCLR. The transfer is made when CPU services the socket routine, therefore is not immediate. HC should wait for this to happen and check for socket status later. The time required can vary from microseconds up to one millisecond; it depends how busy CPU is servicing other tasks. It is fair to poll the status register every 100µs.

Upon enabling the socket SEI takes the following steps:

1. checks if SnPN (Port Number) register > 0. If it is true, proceeds to the next step. If not, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next steps
2. checks if SnIPA (Source IP Addresses) register value is 0.0.0.0. If it is not, proceeds to next step. If it is, it checks if HIPA (Host IP Address) register value is 0.0.0.0. If it is not, it copies its value to SnIPA register and HSM register to SnSM and proceeds to next step. If it is, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next steps
3. checks if assigned received and transmit buffers are in range of total available memory. If they are in range, proceeds to next step; if not, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next step. SEI does not check if the assigned buffers overlap with other assigned buffers. HC must insure the buffers are assigned right
4. initialize the head and tail receive and transmit pointers and updates the SnCS register as follows:
   - SEN, MB, RBE, RSBA, MODE according to SnCSET / SnCCLR
   - writes one to bit field SS (SocketStatus) indicating that the socket is running

From now on HC can receive and send data.

10.3.2 Sending data in UDP Server mode

Data to be sent in a UDP packet must be written to the transmit buffer in one SPI transaction. For this, HC uses command code 13. Each data packet can be addressed to a specific client, therefore a destination IP address and destination port number must be provided each time. This increases the command code 13 header to ten bytes. If bit MB in SnCS register is set, a destination MAC address is needed as well which adds another 6 bytes to the command header. HC can choose to provide the destination MAC address or to leave it for ARP to resolve it. Manually providing it increases speed and lower traffic on the local network. Once the header is clocked in, HC continues with payload data if any. If data length is an odd number, it must be padded with a dummy byte to an even number.

Table 10.3.2-1: Command code 13 header structure when writing data to the transmit buffer in UDP server mode with bit MB not set

Bit offset	Size [bits]	Name	Description
0	6	CC	Command Code: 13
6	1	RW	Read / Write: 0
7	1	RT	Register Type: 0 for socket registers
8	8	SN	Socket Number
16	16	PDL	Payload Data Length
32	32	PIPA	Peer IP Address
64	16	PPN	Peer Port Number

Table 10.3.2-2: Command code 13 header structure when writing data to the transmit buffer in UDP server mode with bit MB set

Bit offset	Size [bits]	Name	Description
0	6	CC	Command Code: 13
6	1	RW	Read / Write: 0
7	1	RT	Register Type: 0 for socket registers
8	8	SN	Socket Number
16	16	PDL	Payload Data Length
32	48	DMA	Destination MAC Address
80	32	PIPA	Peer IP Address
112	16	PPN	Peer Port Number

Before writing data to the transmit buffer, HC have to make sure there is enough space in the buffer. If however HC attempts to send more data that the transmit buffer can hold, data is discarded and the head pointer is not advanced.

When bit MB is not set, SEI writes an 8 byte packet descriptor for each data packet written to the transmit buffer. When bit MB is set, the descriptor length increases to 14 bytes. HC should take this in consideration when evaluating the available space in the buffer.

Data is written to the transmit buffer as it arrives over the SPI bus. Once all data is in, the head pointer is advanced and bit TBE reads zero indicating that the transmit buffer is not empty. If more data is sent than indicated in the command header, the extra data is ignored.

When bit MB is not set, the socket logic proceeds in the same way when in UDP client mode.

Fig. 10.3.2-1: Example: Write 3 bytes of data to socket 2 transmit buffer in UDP Server mode with bit MB not set

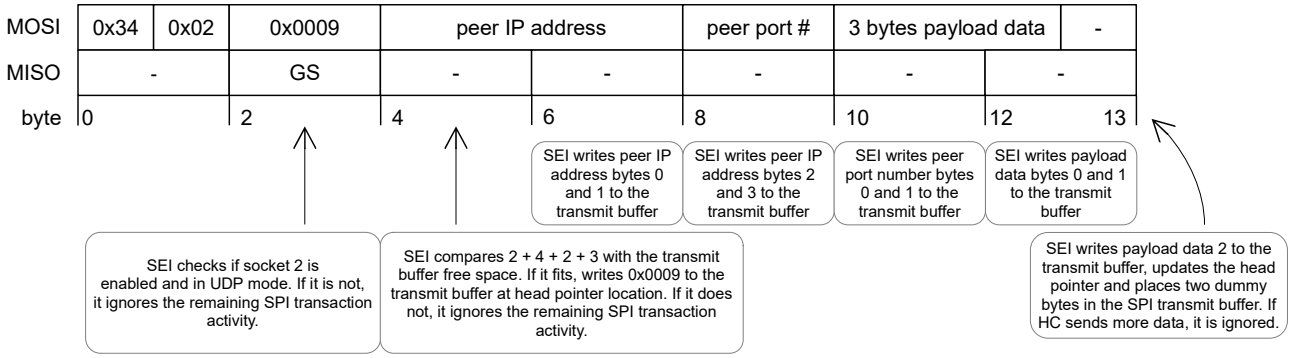
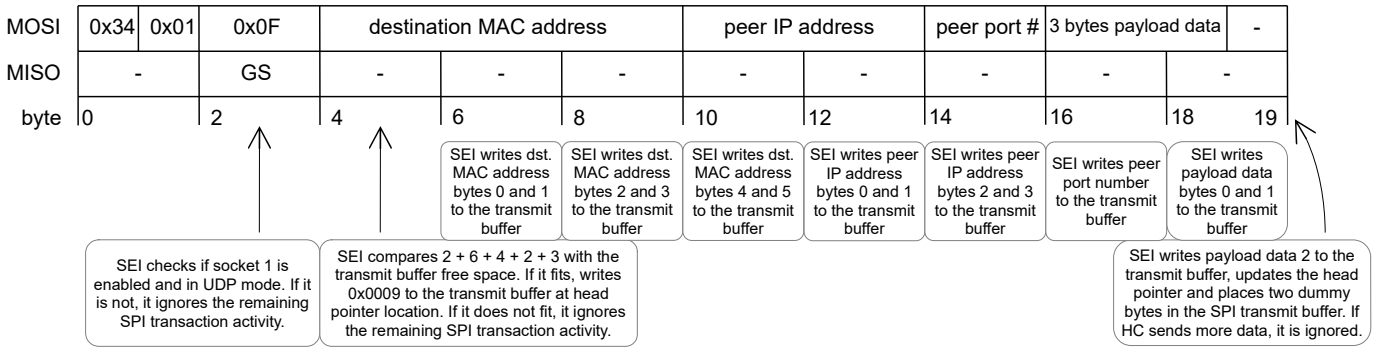


Fig. 10.3.2-2: Example: Write 3 bytes of data to socket 1transmit buffer in UDP Server mode with bit MB set



Once the packet is sent, SEI advances the tail pointer. If the transmit buffer gets empty, bit TBE in SnCS register will read one indicating that the transmit buffer is empty.

10.3.3 Receiving data in UDP Server mode

For a UDP packet to be received by a socket in UDP Server mode, the following conditions must be true:

– packet destination IP address equals SnIPA register (local IP address) or if bit RSBA in SnCS register is set a subnet broadcast IP destination is accepted or if bit RBA in SnCS register is set, a broadcast IP address (FF.FF.FF.FF) is accepted
– packet destination port number equals SnPN register (local port number)
– there is enough space in the receive buffer to store the packet descriptor and payload data. If there is not enough space in the receive buffer the received packet is lost and bit SnIF in IF register is set. If the pin interrupt is enabled, an interrupt is issued.

If the conditions above are met, SEI takes the following actions:

– copies packet source port number to SnPPN (Peer Port Number) register
– writes a packet descriptor to the receive buffer
– copies the payload data from the Ethernet buffer to the receive buffer
– advances the head pointer
– it sets bit SnIP in IF register. If the interrupt pin is enabled, an interrupt is issued.

HC reads data using command code 13. A whole received data packet must be read at once. This command uses a two byte header. SEI first places two dummy bytes on the MISO line followed by the GC register and then a 16 bit value that indicates the length of the following packet descriptor and data. If bit MB in SnCS register is set, SEI places the received UDP packet source MAC address on the MISO line. If bit MB is not set, SEI continues with the peer IP address, peer port number and finally the payload data. If the received buffer is empty and HC attempts to read a packet, the length field reads 0xFFFF. If data length is an odd number, HC must clock in an extra byte to make the total even.

Once the entire packet data is read, SEI advances the tail pointer. If HC keeps reading more data during the same SPI transaction, data read is not valid.

HC should keep reading more data packets until empties the receive buffer in order to avoid overflow.

Command code 13 header structure when reading data from the receive buffer is the same with the one in UDP Client mode

Table 10.3.3-1: Command code 13 header structure when reading data from the receive buffer (UDP Server mode)

Bit offset	Size [bits]	Name	Description
0	6	CC	Command Code: 13
6	1	RW	Read / Write: 1
7	1	RT	Register Type: 0 (socket registers)
8	8	SN	Socket Number

Table 10.3.3-2: UDP Server mode read data structure with bit MB in SnCS register not set

Byte offset	Size [bytes]	Name	Description
0	2	-	Undefined, any, dummy bytes
2	2	GS	General Status register
4	2	PDL6	Payload Data Length + 6
6	4	SIPA	received UDP packet Source IP Address
10	2	SPN	received UDP packet Source Port Number
12	PDL*	PD	Payload Data

* if PDL is an odd number, add an extra undefined byte

Fig. 10.3.3-1: Example: Read a received UDP packet of 5 bytes from socket 3 in UDP Server mode when bit MB in SnCS register is not set

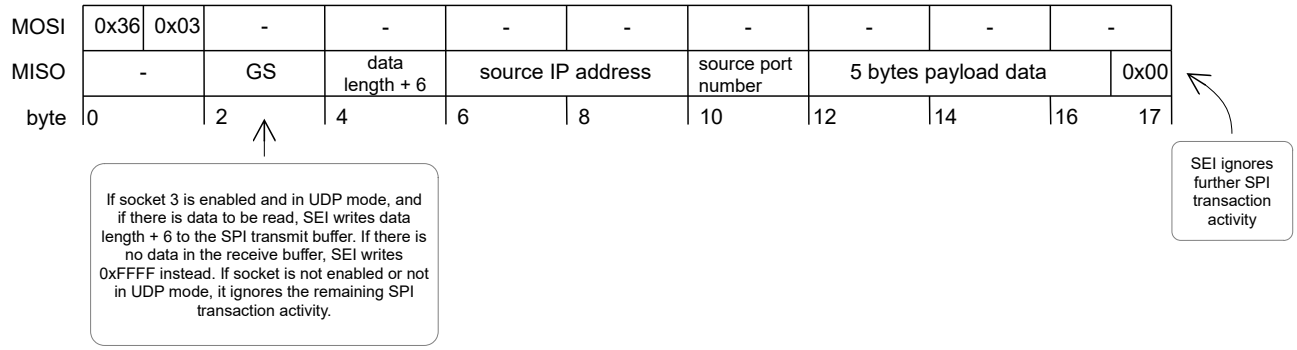
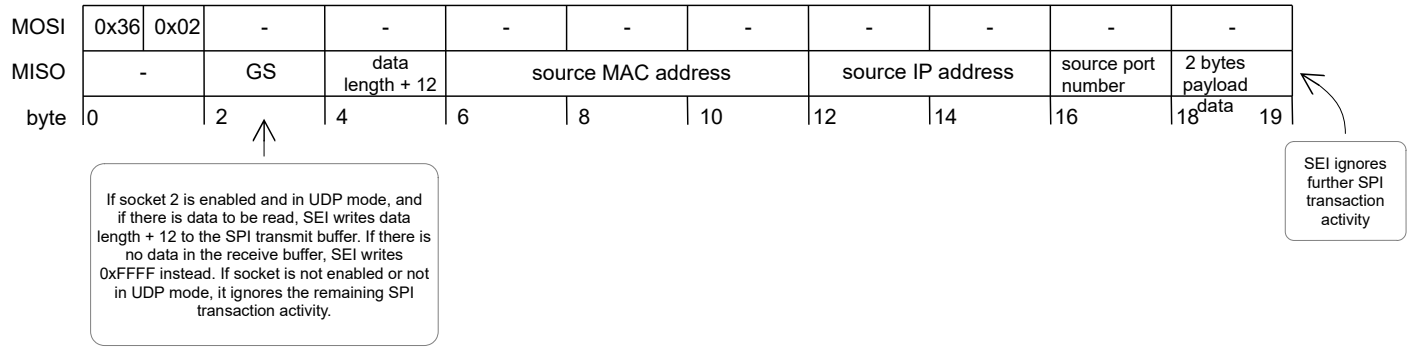


Table 10.3.3-3: UDP Server mode read data structure with bit MB in SnCS register set

Byte offset	Size [bytes]	Name	Description
0	2	-	Undefined, any, dummy bytes
2	2	GS	General Status register
4	2	PDL6	Payload Data Length + 12
6	6	SMAC	received UDP packet Source MAC Address
12	4	SIPA	received UDP packet Source IP Address
16	2	SPN	received UDP packet Source Port Number
18	PDL*	PD	Payload Data

* if PDL is an odd number, add an extra undefined byte

Fig. 10.3.3-2: Example: Read a received UDP packet of 2 bytes from socket 2 in UDP Server mode when bit MB in SnCS register is set



10.3.4 Disable socket in UDP Server mode

HC can disable the socket by simply writing 0xFFFF to the SnCCLR register. This operation is handled by the socket routine therefore if needed, HC should check for its completion.

When SEI disables the socket, if performs the followings:

– clears the first 12 bits in SnCS register
– clears register SnPN (local Port Number)

A socket requires the least CPU intervention when is disabled, therefore HC should keep unused sockets disabled.

11 TCP – Transport Control Protocol

11.1 TCP Overview

Unlike in case of UDP, HC does not have control about how data is packed and sent over the network. First, both parties establish a connection before data can be exchanged. Data is exchanged at byte level and not at packet level. When done, the connection is closed.

SEI provides two TCP modes: client and server.

The client side initiate a connection. The server side listen and accepts a connection. When done, either part can initiate the connection closing process.

11.2 Enable Socket in TCP Client mode

To enable a socket in TCP Client mode, follow the next steps:

1. write source IP address to SnIPA register if different than HIPA
2. write subnet mask to SnSM register if different than HSM
3. write peer IP address to SnPIPA register
4. if a fixed peer MAC address is needed, write it to SnPMAC register. This writes a static row in the ARP table with the SnPIPA in the IP field and SnPMAC in the MAC field. Otherwise skip this step and ARP will attempt to resolve the destination MAC address.
5. if a specific local port number is preferred, write it to SnPN register; if not, leave it as zero and an ephemeral port will be assigned before the SYN packet is sent
6. write the peer port number to SnPPN register
7. assign a receive and transmit buffer by writing to SnBSPS; make sure these buffers does not overlap with other enabled socket buffers
8. optionally, HC can write the SnTTL with a time-to-live value. The default value is 128.
9. optionally HC can write to SnSMTU register if other value than 1500 is needed; the receiving MTU is always 1500 bytes
10. enable the socket by writing the following fields in SnCSET register in one go:
    - write 2 to bit field MODE
    - set bit TOC (TCP Open Close) if a connection is to be made right away; alternatively, this bit can be set later
    - write 1 to SEN (SocketEnable) bit

The Socket Control and Status register SnCS can only be set and clear by writing to SnCSET and SnCCLR. The transfer is made when CPU services the socket routine, therefore is not immediate. HC should wait for this to happen and check for socket status later. The time required can vary from microseconds up to one millisecond and it depends on how busy CPU is servicing other tasks. It is fair to poll the status register every 100µs.

Upon enabling the socket in TCP Client mode, SEI takes the following steps:

1. checks if SnPPN (Peer Port Number) register is not zero. If it is, the socket is enabled with Configuration Error status (15) and skips the next steps. The only way to get out from this state is to disable the socket.
2. checks if source and destination IP addresses are in the same subnet
   - if yes, the destination IP address is the one that ARP will be required to get a MAC address for
   - if not, it checks if the gateway IP address is in the same subnet; if yes, the gateway IP address is the one ARP will be asked to get a MAC address for; if not, the socket is enabled with Configuration Error status (15) and skips the next steps. The only way to get out from this state is to disable the socket.
3. checks if assigned received and transmit buffers are in range of total available memory. If they are in range, proceeds to next step; if not, the socket is enabled with Configuration Error status (15) and skips the next steps. The only way to get out from this state is to disable the socket. SEI does not check if the assigned buffers overlap with

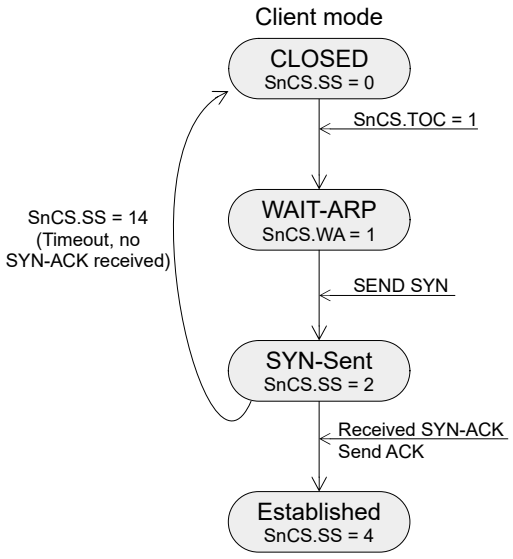
other assigned buffers. HC must insure the buffers does not overlap

4. updates the SnCS register as follows:
   - clears bit field SS (SocketStatus) indicating that the connection is closed
   - sets SEN bit indicating that the socket is enabled
   - sets bit field MODE to 2 indicating that the socket is in TCP Client mode

At this point the socket is enabled but no connection established. The socket does not accept data from HC yet. The transmit and receive buffer pointers are not initialized yet.

11.3 TCP Client Connect

Fig. 11.3-1: TCP client connection diagram



In order for the socket to connect to its peer, HC sets bit TOC in SnCS register.

Once bit TOC is set, SEI takes the following steps:

1. if local port number (register SnPN) is zero, it assigns an ephemeral local port number
2. it initializes the receive and transmit pointers
3. it sets SnPMSS to 536
4. it clears bit CFR (Close on FIN Received) in SnCS register
5. it looks in the ARP table for a row with a status 'acquired' or 'static' and the IP field equal to the one in SnPIPA register. If found, it skips the next step
6. it fills the IP field in the associated row in the ARP table with the destination IP address and sets the status as “Acquiring”. If the acquiring process is successful, it proceeds to next step. If not, field SS in SnCS register changes to “Timeout” and an interrupt is issued.
7. it sends a SYN packet
8. it initializes bit field SS to 2 (SYN_SENT)

SEI runs a timer that expires after a RTO (retransmit timeout) period. SEI initializes this time to one second. If no SYNACK packet is received, the SYN packet is resent and RTO doubled and limited to 60 seconds. SEI retransmits the SYN packet up to a number of times in SnTCPT register.

If no SYNACK is received, socket enters state Timeout (SS = 14). The socket remains in this state until HC take action like to disable the socket or try to reconnect.

If a RESET packet is received in response to the SYN packet, socket enters state RESET-BY-PEER (SS = 13). The socket remains in this state until HC take action like to disable the socket or try to reconnect.

If a SYNACK packet is received, SEI updates register SnPMSS if option-kind 2 is received.

SEI does not use window scaling even though a socket receive buffer can be larger than 64kB. When the receive buffer is larger than 64kB, it advertise a receive window of 65535 bytes.

SEI enters state ESTABLISHED and sets bit field SS to 4.

SEI transmits an ACK packet.

11.4 Enable Socket in TCP Server mode

To enable a socket in TCP Server mode, follow the next steps:

1. write source IP address to SnIPA register; usually this is the same with the host IP address in register HIPA
2. write subnet mask to SnSM register
3. write the local port number the server is to listen to SnPN register
4. enable the socket by writing the following fields in SnCSET in one go:

   - write 1 to IPDF bit if “Do not fragment” flag in the Ipv4 header is to be set. For highest performance keep this bit set

   - write 3 to bit field MODE

   - set bit TOC (TCP Open Close) if the socket is to enter “Listening” state right away; alternatively, this bit can be set later

   - write 1 to SEN (SocketEnable) bit

The Socket Control and Status register SnCS can only be set and clear by writing to SnCSET and SnCCLR. The transfer is made when CPU services the socket routine, therefore is not immediate. HC should wait for this to happen and check for socket status later. The time required can vary from microseconds up to one millisecond; it depends how busy CPU is servicing other tasks. It is fair to poll the status register every 100µs.

Upon enabling the socket in TCP Server mode, SEI takes the following steps:

1. if SnPN (local Port Number register) = 0, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next steps
2. if SnIPA (local IP address) = “0.0.0.0”, and HIPA = “0.0.0.0”, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next steps
3. checks if assigned received and transmit buffers are in range of total available memory. If they are in range, proceeds to next step; if not, the socket will not be enabled, writes Configuration Error status value 15 to SS field in SnCS register and skips the next step. SEI does not check if the assigned buffers overlap with other assigned buffers. HC must insure the buffers are assigned right
4. updates the SnCS register as follows:

   - clears bit field SS (SocketStatus) indicating that the connection is closed

   - sets SEN bit indicating that the socket is enabled

   - sets bit field MODE to 2 indicating that the socket is in TCP Server mode

At this point the socket is enabled but no connection established. The socket does not accept data from HC. The transmit and receive buffer pointers are not initialized.

11.5 TCP Server Listening

A socket in TCP Server mode accepts a connection when its status is “LISTENING” (SS = 1). To place a socket in TCP Server mode from Closed state to Listening, HC sets bit TOC in SnCS register

Once bit TOC is set, SEI takes the following steps:

1. initializes SnPMSS register to 536 bytes
2. initializes receive and transmit buffer pointers to zero
3. initializes bit field SS to 1 Listening

A socket in TCP mode and Listening status can only accept one connection. If SEI is supposed to accept multiple concurrent connection, multiple sockets must be set in TCP Server mode.

At this point, the socket waits for a client to send a SYN packet. When a SYN packet is received, SEI takes the following actions:

1. updates SnPIPA register from the received packet
2. updates SnPPN register from the received packet
3. updates SnPMTU register if option-kind 2 is received
4. clears bit CFR (Close on FIN Received) in SnCS register
5. enters SYN-RECEIVED state SS = 3
6. sends a SYN_ACK packet

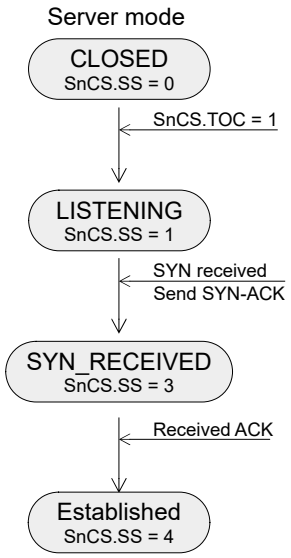
SEI expects an acknowledgement in response to the SYN-ACK packet sent. SEI runs a timer that expires after a RTO (retransmit timeout) period. SEI initializes this time to one second. If no ACK packet is received, the SYN-ACK packet is resent and RTO doubled and limited to 60 seconds. SEI retransmits the SYN-ACK packet up to a number of times in SnTCPT register.

If no ACK packet is received, socket enters state TIMEOUT (SS = 14). The socket remains in this state until HC take action like to disable the socket or put the socket in Listening mode again.

If a RESET packet is received in response to the SYN-ACK packet, socket enters state RESET-BY-PEER (SS = 13). The socket remains in this state until HC take action like to disable the socket or put the socket in Listening mode again.

If an ACK packet is received, it enters ESTABLISHED state and SS bit field updated to 4.

Fig. 11.5-1: TCP server listening and accepting a connection diagram



11.6 TCP in Established state

The established state is when HC exchanges data with its peer and it works the same way whether the socket is in client or server mode. It can remain in this state for as long as needed.

11.7 Sending data in TCP mode

Once the connection is established, HC can send data writing to register SnDT (Socket Data Tcp) using command code 14. Unlike in UDP mode, HC sends data as it needs, without to worry or having control how is packetized.

Table 11.7-1: Command code 14 header structure when writing data to the transmit buffer in TCP mode

Bit offset	Size [bits]	Name	Description
0	6	CC	Command Code: 14
6	1	RW	Read / Write: 0
7	1	RT	Register Type: 0 (socket register)
8	8	SN	Socket Number
16	1	PUSH	PUSH bit
17	1	HOLD	HOLD bit
18	14	LEN	Length of data that follows

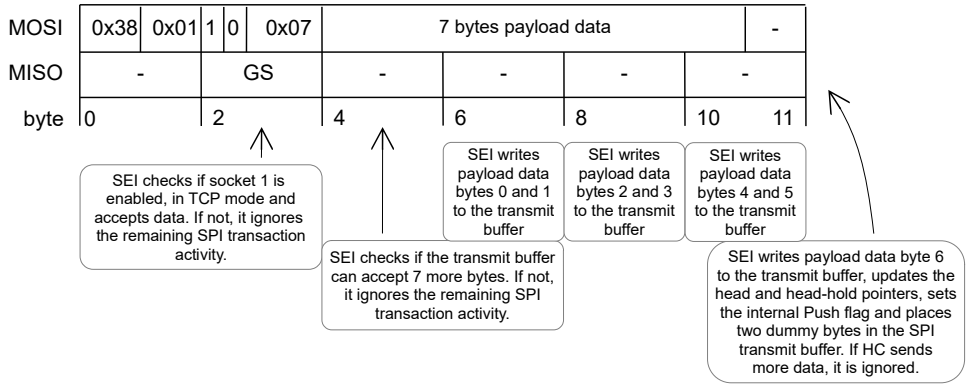
SEI maintains a bit flag SendPush. When TCP header is built, this flag is copied to TCP header PSH position. There is no mechanism to associate the timing HC sets this bit PUSH with the data sequence number. If HC wants to attach the PSH flag in the TCP header with the sequence number, it have to do so when the transmit buffer is empty, send data with bit PUSH set, and wait until the transmit buffer becomes empty again.

To help a HC with limited amount of RAM and to send data over the network in larger packets, HC can write data to the transmit buffer and hold it until more data is written. For this purpose SEI maintains two transmit head pointers that can be seen by reading registers SnTHP (Transmit Head Pointer) and SnTHHP (Transmit Head Hold Pointer). HC can hold data written by writing a one to bit HOLD. When HC writes data with bit HOLD set, SEI advances only the Transmit Head Hold Pointer. When bit Hold is not set, both pointers are advanced and made equal. When all data is written, HC clears this bit with the last data written. The TCP routine transmits data in between Transmit Tail Pointer and Transmit Head Pointer.

HC can write to the transmit buffer up to 16383 bytes at once. If however, HC attempts to send more data that the transmit buffer can hold, data is discarded and the head pointer is not advanced.

Data is written to the transmit buffer as it arrives over the SPI bus. Once all data is in, SEI advances the head pointer and bit TBE reads zero indicating that the transmit buffer is not empty. Data is sent out once the socket gets serviced by CPU and the TCP protocol permits. If more data is sent than indicated in the command header, the extra data is ignored.

Fig. 11.7-1: Example: Write 7 bytes of data to socket 1 transmit buffer in TCP mode, flag PUSH set, and send data right away



11.8 Receiving data in TCP mode

When data arrives in the receive buffer, SEI issues an interrupt. When there is at least one byte in the receive buffer, bit DRB reads 1.

When reading data from the receive buffer, HC can read up to 65535 bytes, providing data is available. HC can also read the SnRDS register to find out how much data is in the receive buffer.

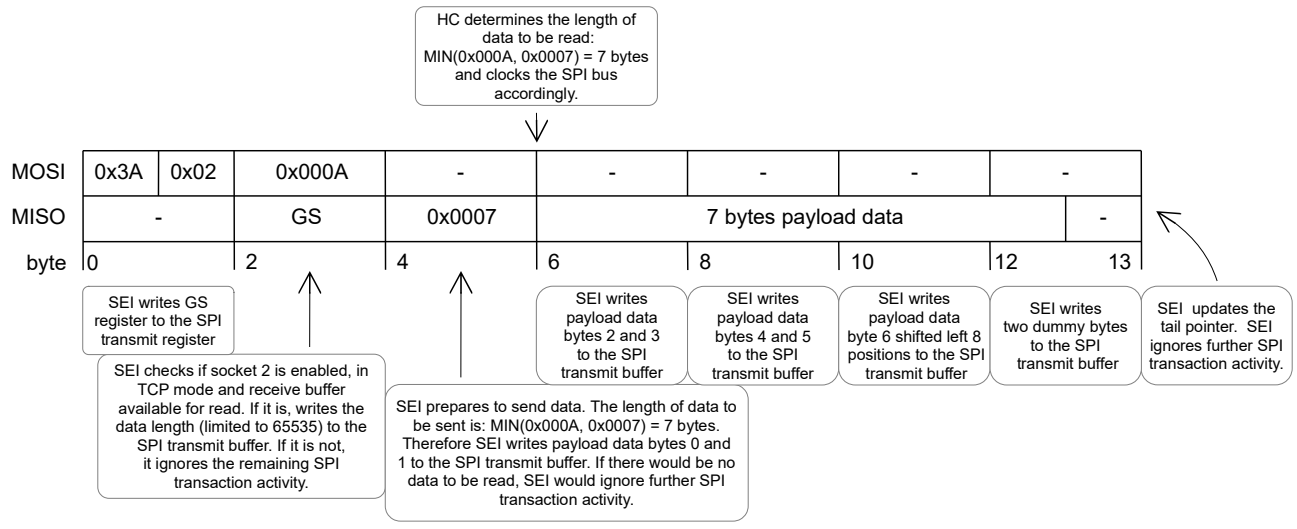
Another method is to just read register SnDT. Command code 14 provides a mechanism that HC can use to read data without knowing in advance about the amount of data in the buffer.

Table 11.8-1: Command code 14 when reading data from the receive buffer (TCP mode)

Bit offset	Size [bits]	HC - MOSI		SEI - MISO	
		Name		Name	
0	6	CC	Command Code: 14	U	Undefined
6	1	RW	Read / Write: 1		
7	1	RT	Register Type: 0		
8	8	SN	Socket Number		
16	16	DR	Data Read Length	GS	General Status register
32	1	U	Two dummy bytes	U	Undefined
33	15			DA	Data Available Length
48	-	U	MIN(DR, DA) of dummy bytes*	PD	MIN(DR, DA) Payload Data*

* if MIN(DR, DA) is an odd number, add an extra dummy byte

Fig. 11.8-1: Example: HC is willing to read up to 10 bytes from socket 2 receive buffer while only 7 are in there. HC ends up reading only 7 bytes



11.9 TCP Slow Start and Congestion Avoidance

In TCP Client mode, just before sending the first SYN packet, and in TCP Server mode before responding with a SYN-ACK, SEI initializes SnRTO (Retransmit Timeout) register to 200, which in 5ms units amounts to one second.

Once the connection is established, SEI sets the socket slow start threshold to:

$$SnSST = \frac{peer\ window\ \text{size}}{SnPMSS}$$

rounded to the lower integer.

If the SYN or SYN-ACK packets were not retransmitted, SEI sets the congestion window to 4 for SnPMSS <= 1095 or 3 otherwise.

If the SYN or SYN-ACK packets were retransmitted, SEI sets the congestion window to 1.

Registers SnSST and SnCW are kept in units of SnPMSS.

SEI maintains an internal variable called SnCwBytesCnt which counts the number of bytes acknowledged by the other party.

When SnCW is less than SnSRTT, the algoritm is in Slow Start mode. In this mode, with each new data acknowledged, SEI performs the following:

```
SnCwBytesCnt =  SnCwBytesCnt + new data acknowledged
if  SnCwBytesCnt => SMSS
        SnCW = SnCW + 1
         SnCwBytesCnt = SnCwBytesCnt - SMSS
if  SnCwBytesCnt => SMSS
        SnCW = SnCW + 1
         SnCwBytesCnt = 0
```

where SMSS is the Sender Maximum Segment Size and is calculated as follow:

SMSS = MIN(SnPMSS, SnMTU – 40)

When SnCW equals or exceeds SnSST, the algorithm enters the Congestion Avoidance mode. In this mode, with each new data acknowledged, SEI performs the following:

```
SnCwBytesCnt =  SnCwBytesCnt + new data acknowledged
if  SnCwBytesCnt => SnCW * SMSS
        SnCW = SnCW + 1
         SnCwBytesCnt = 0
```

11.10 TCP Retransmit Timeout calculation

SEI continuously adjusts SnRTO register based on network conditions. To calculate SnRTO, SEI uses two internal variables: SRTT and RTTVAR, and the algorithm described in RFC6298 with some exceptions:

1. SEI initializes SnRTO to one second as described in RFC6298 page2 paragraph 2.1
2. instead of calculating SRTT and RTVAR as described in RFC6298 page2 paragraph 2.2, SEI initializes SRTT and RTTVAR to 200ms. This results in a SnRTO of one second as initialized
3. SEI uses the same formula described in RFC6298 page2 paragraph 2.3 to calculate SRTT and RTTVAR once a round-trip-time measurement has been made.
4. SEI limits the SnRTO value to the one in register SnTCPT which is initialized to one second and is the same with the one recommended by RFC6298 page 2 paragraph 2.4. However, SEI gives HC the possibility to lower this value. Lower values may be more appropriate when both SEI and the other party are in the same LAN or even in the same metropolitan area network where round-trip-time is often in the order of few milliseconds.

After sending a packet with data, SEI waits for an acknowledgement until the retransmit timeout timer expires. If it is the first time the packet is to be resent, SEI also updates the congestion window and the slow start threshold registers as follow:

SnCW = 1

SnSST = MAX(data sent but not acked / 2, 2)

SnCW = 1 suggests that SEI sends one packet from the beginning of the congestion window and waits for an ACK. SEI retransmits the packet up to a number in SnTCPT. If an ACK packet is not received at all, the connection is silently closed, SS field updated to 14 – “TCP Timeout” and an interrupt issued. HC can still read data if available in the receive buffer. In this stage, HC can try a new connection in case of client mode or put the socket in Listening mode if in server mode.

11.11 TCP Fast Retransmit

When a particular sequence number is consecutively acknowledged four times with packets without data and the peer does not advertise a new receive window, SEI retransmits a packet starting with that particular sequence number without waiting for the retransmit timeout timer to expire and updates SnCW and SnSST as follows:

SnCW = 1

SnSST = MAX(data sent but not acked / 2, 2)

SnCW = 1 suggests that SEI sends one packet from the beginning of the congestion window and waits for an ACK which is possible that acknowledges more than one packet. This is a special case when first ACK is received after the retransmitted data. In this case, the SnSST is increased with the number of packets acknowledged.

SEI does not buffer data received if there is a gap in between.

11.12 TCP Zero Window Probe

When HC reads data not fast enough, there is a chance that the receive buffer fills up. SEI therefore advertises a zero receive window. If the other party wants to send more data, it may periodically send a “zero window probe” just to get back an ACK which reveals SEI's receive window size. SEI responds with a due acknowledgement. In this state, if HC reads data, SEI immediately sends a packet advertising a non zero receive window. In this circumstances HC should read as much data as possible at once in order to avoid receiving small packets of data at a high rate. In the mean time, HC can send data as usual.

When peer's window reads zero and SEI has data in the transmit buffer to send, it starts a timer with the initial period equal to the value in register SnRTO. When this timer expires, SEI send a zero window probe. If the response shows that the peer's receive window is still zero, it doubles the timer period – limited to one minute – and starts the timer. This process continues until the peer's receive window becomes non zero. In the meant time, HC can receive data from its peer. When data arrives, it carries information about peer's receive window and if still shows as zero, SEI restarts the sending zero window probe process again with an initial period equal to the value in SnRTO register.

11.13 TCP Keep Alive

A TCP connection does not have a time limit. In the same time, a TCP connection requires resources and maintenance. Servers are often required to accept many connection from multiple clients. Routers and gateways are also required to keep information about ongoing connections.

Servers often send a Keep Alive packet to inform routers along the path that the connection is still alive and to probe a client status. If the client responds, then the server keeps the connection. If there is no response or a packet with the bit RST flag set is received, the server releases the resources for the existing connection in order to be used for another connection.

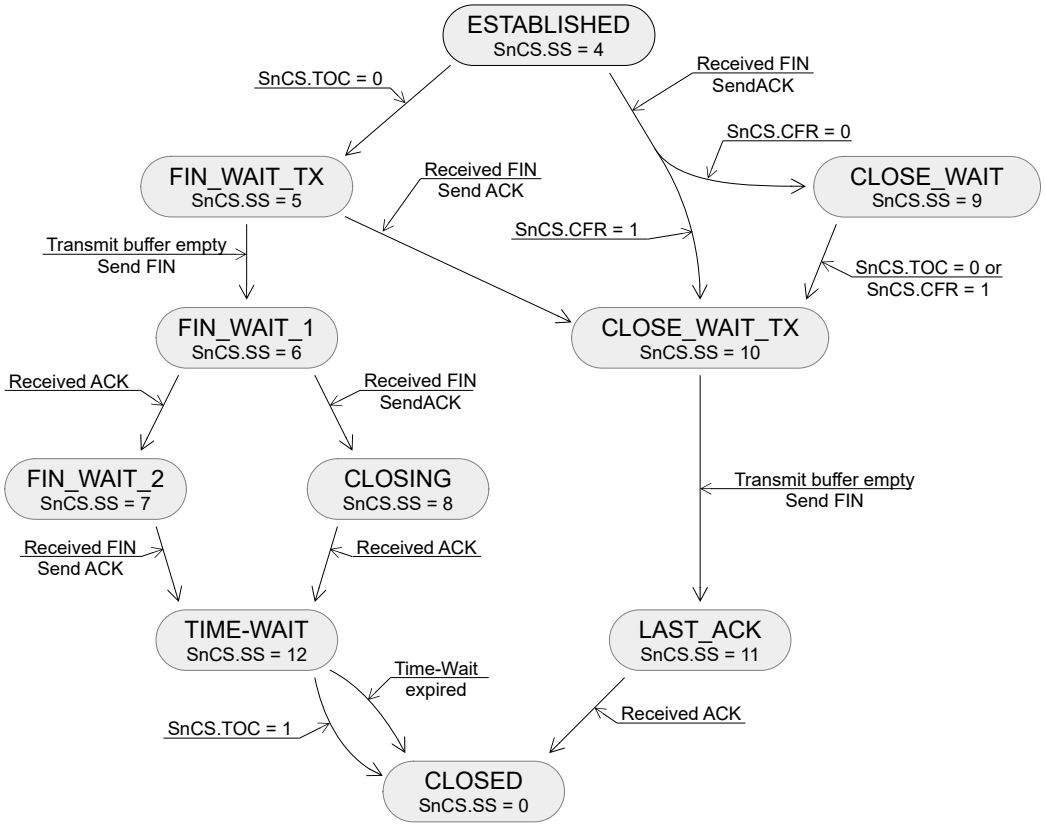
The Keep Alive algorithm is enabled by default and the keep alive timer is set in units of 30s multiplied by the KPA field value in SnTCPT register. A value of zero disables the algorithm.

When the keep alive timer expires, three keep alive probes are sent at one second interval. If no response is received, the connection is silently closed, SS field updated to 14 – “TCP Timeout” and an interrupt issued.

11.14 TCP Closing Connection

When one of the party has no more data to send and the application no longer need the connection, it proceeds to close it. In the process, the connection goes through a number of intermediate steps before reaches the Closed status.

Fig. 11.14-1: TCP closing connection state diagram



ESTABLISHED. HC can initiate the connection closure by clearing bit TOC in SnCS register. By doing that, the connection enters FIN_WAIT_TX state, SEI updates SS field to 5 and no more payload data to be sent is accepted from HC.

In case the other party closes the connection, SEI receives a FIN packet. If bit CFR in SnCS register is set, connection enters CLOSE_WAIT_TX state, SS is updated to 10 and SEI no longer accepts payload data from HC. If bit CFR is not set, connection advances to CLOSE_WAIT state, SS is updated to 9.

FIN_WAIT_TX. The connection stays in FIN_WAIT_TX state until all data in the transmit buffer is sent and acknowledged. Once the transmit buffer is empty, SEI sends a FIN packet, updates SS filed to 6 and the connection reaches FIN_WAIT_1 status.

If while data in the transmit buffer is sent and acknowledged a FIN packet is received, the connection enters CLOSE_WAIT_TX state.

FIN_WAIT_1. If an acknowledgement packet to the previous FIN packet sent arrives, the connection enters state FIN_WAIT_2 and SS field is updated to 7.

If a FIN is received, the connection enters state CLOSING with SS = 8.

FIN_WAIT_2. SEI waits for the other party to close the other half of the connection. In this state SEI can still receive data and HC can read it from the receive buffer. When a FIN packet is received, it replies with an ACK, connection enters state TIME-WAIT and SS is updated to 12.

CLOSING. SEI awaits for an ACK in response to the FIN packet sent. Once the FIN packet is acknowledged, the connection enters state TIME-WAIT and SS field is updated to 12.

TIME-WAIT. The connection stays in this state for up to four minutes. This is needed in order for any lost packets to leave the network. HC has two choices: one is to simply wait for the timer to expire and the connection to reach the status CLOSED, or to force the socket to abandon the current connection and open a new one if in client mode or to set the socket in LISTENING mode if in server mode.

To force a new connection, HC sets bit TOC in SnCS register. If the socket is in client mode, SEI sends a new SYN packet and the connection advances to SYN_SENT state and SS updates accordingly.

If the socket is in server mode, the connection closes and the socket enters the LISTENING state.

CLOSE_WAIT. In this state a FIN has been received, an ACK has been sent out. HC can keep sending data. When done, HC clears bit TOC or sets bit CFR in SnCS register. HC can do this even before all data as been sent out and acknowledged. However, the socket does not accept new payload data from HC. The connection enters state CLOSE_WAIT_TX and SS is updated to 10.

CLOSE_WAIT_TX. The connection is in this state until all data in the transmit buffer is sent and acknowledged. Once done, SEI sends a FIN packet, updates bit field SS to 11 and the connection enters state LAST_ACK.

LAST_ACK. In this state SEI waits for an ACK packet that would acknowledge the FIN send before. When the AC packet arrives, the connection is closed and bit field SS updated to zero.

12 Socket registers detailed description

Legend:
- R – readable bit
- W – writable bit
- U – unimplemented
- n – value at POR
- u – value unknown
- 0 – bit cleared
- 1 – bit set

SnCS

Socket Control and Status register. Command code: 0. Size: 2 bytes. This register is read only. Write to this register using registers SnCSCLR and SnCSSET.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	R-0	U-0	R-1	R-0	R-0			
	PDPIF	WA	TBE	DRB	SS			
0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0			
	SEN	MB	CFR/RBE	TOC/RSBA	MODE			

bit 15 **PDPIF:** Packet Dropped / Push Interrupt Flag. When in UDP mode, this bit is set if a packet is received and there is not enough space for it in the receive buffer. When in TCP mode, this bit is set when a packet is received with the PUSH flag set. HC clears this bit by reading this register.

bit 14 **WA:** Wait for ARP. This bit is set while the socket waits for an ARP resolution.

bit 13 **TBE:** Transmit Buffer Empty. This bit reads one when the transmit buffer is empty, zero otherwise

bit 12 **DRB:** Data in Receive Buffer. This bit reads one when the receive buffer is not empty, zero otherwise

bit 11÷8 **SS:** Socket Status

SS	UDP	TCP
0	CLOSED	CLOSED
1	RUNNING	WAIT for ARP
2		SYN-SENT (client mode) LISTENING (server mode)
3		SYN-RECEIVED (server mode)
4		ESTABLISHED
5		FIN-WAIT-Tx
6		FIN-WAIT-1
7		FIN-WAIT-2
8		CLOSING
9		CLOSE-WAIT
10		CLOSE-WAIT-Tx
11		LAST-ACK
12		TIME-WAIT
13		RESET-BY-PEER
14	ARP TIMEOUT	ARP TIMEOUT / TCP TIMEOUT
15	CONFIGURATION ERROR	

bit 7 **SEN:** Socket Enable.
0: disable socket
1: enable socket

bit 6 **MB:** MAC present in data buffer. This bit is relevant only when socket is in UDP Server mode. When this bit is set, the sender MAC address is present in the packet descriptor. It follows the two bytes for packet length and is followed by the sender's IP address.

bit 5 **CFR:** TCP Close on FIN Received. By default, when peer device sends a FIN packet, an ACK is sent, SnSCIF bit in IF register is set indicating to the host controller that the connection changed status and SS bit field updated with the new socket status. The host controller can choose to complete the connection closure or to continue to send more data. When this bit is set and a FIN packet is received, the socket logic clears bit TOC and proceeds to close the connection. By setting this bit the socket logic no longer accepts new payload data from the host controller. This bit cannot be set when the connection is in a CLOSED state. The socket logic clears this bit when the connection advances from CLOSED state onward.
RBA: Receive Broadcast Allowed, used in UDP Server mode. When this bit is set, socket also accepts UDP packets with a destination IP address of 255.255.255.255.

bit 4 **TOC:** TCP Open / Close connection
When in TCP mode:
- set this bit in order to open a connection or put the socket in LISTENING mode. This bit can be set only when the connection is in a CLOSED or TIME-WAIT state.
- clear this bit in order to close a connection. This bit can be cleared at any time. After clearing this bit, the socket logic no longer accepts new payload data from the host controller. The closing procedure starts only after all data in the transmit buffer has been sent and acknowledged. Check the status of the connection (bits 11-8) in order to find out when the connection is actually closed.
To close a connection by sending a RST packet, clear bits CFR and TOC at the same time (by writing 0x30 to SnCSCLR register). This only works when the connection is in a state between SYN-RECEIVED and LAST-ACK.
RSBA: Receive Subnet Broadcast Allowed, used in UDP Server mode. When this bit is set, socket also accepts UDP packets with the subnet broadcast IP address.

bit 3-0 **MODE:** Socket mode. These bits can only be written when the socket is disabled.
b0000: UDP Client
b0001: UDP Server
b0010: TCP Client
b0011: TCP Server
The remaining values are not implemented.

SnCSCLR

Socket Control and Status Clear register. Command code: 0. Size: 2 bytes. This register is write only. Use this register to clear specific bits in SnCS register.

W	W	W	W
byte 1	byte 2	byte 3	byte 4

SnCSSET

Socket Control and Status Set register. Command code: 1. Size: 2 bytes. This register is write only. Use this register to set specific bits in SnCS register.

W	W	W	W
byte 1	byte 2	byte 3	byte 4

SnPMA

Socket Peer MAC Address register. Command code: 2. Size: 6 bytes. This register only provides a way to write a row in the ARP table. It does not exists as a socket register. Register SnIPA must be non zero in order for this operation to succeed. The row number written is the same with te socket number used for this SPI write. The row in ARP table becomes static, meaning that it does not expire. To clear the ARP row or in other words to make it “empty”, write zero to this register. When read, this register returns the MAC address in the ARP table in the row assigned to this socket. If the assigned row is “empty”, six bytes of zero are returned.

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4	byte 5	byte 6

SnIPA

Socket IP Address register. Command code: 3. Size: 4 bytes. This register holds the Socket IP address. Following a reset, the value of this register is 0.0.0.0. The host controller can write to this register only when the socket is disabled.

R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4

SnSM

Socket Subnet Mask register. Command code: 4. Size: 4 bytes. This register holds the Socket Subnet Mask. Following a reset, the value of this register is 0.0.0.0. The host controller can write to this register only when the socket is disabled.

R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4

SnPIPA

Socket Peer IP Address register. Command code: 5. Size: 4 bytes. This register holds the peer IP address. Following a reset, the value of this register is 0.0.0.0. HC can write to this register only when the socket is disabled. In TCP Server mode the socket logic writes this register with the peer IP address once the first SYN packet is received.

R/W-0	R/W-0	R/W-0	R/W-0
byte 1	byte 2	byte 3	byte 4

SnP

Socket Port number register. Command code: 6. Size: 2 bytes. SPI interface handles this register in little-endian format. This register can be written only when the socket is disabled. Following a reset, the value of this register is 0.

UDP Client mode: if this register value is zero when the socket is enabled, an ephemeral port number is assigned just before the first Ethernet packet is about to be sent out. Socket logic clears this register when socket gets disabled.

TCP Client mode: if this register value is zero when a connection is initiated, an ephemeral port number is assigned just before the first SYN packet is about to be sent out. Socket logic clears this register after the socket gets disabled.

UDP server mode: the socket can be enabled only if this register is non zero.

TCP server mode: the socket can be enabled only if this register is non zero.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	SnP1<7:0>							
7÷0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	SnP0<7:0>							

bit 15÷8 **SnP1<7:0>:** Socket Port number byte 1

bit 7÷0 **SnP0<7:0>:** Socket Port number byte 0

SnPP

Socket Peer Port number register. Command code: 7. Size: 2 bytes. SPI interface handles this register in little-endian format. This register holds the socket peer port number. Following a reset, the value of this register is 0. HC can write to this register only when the socket is disabled.

Bit Range	Bit 15/7	Bit 14/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0
15÷8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	SnPP1<7:0>							
7÷0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	SnPP0<7:0>							

bit 15÷8 **SnPP1<7:0>:** Socket Peer Port number byte 1

bit 7÷0 **SnPP0<7:0>:** Socket Peer Port number byte 0

SnTTL

Socket Time To Live register. Command code: 8. Size: 2 bytes. The socket logic writes the upper byte of this register to the IP TTL field when it sends out an IP packet.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	R/W-128							
	TTL<7:0>							
7÷0	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	-	-	-	-	-	-	-	-

bit 15÷8 **TTL:** IP Time To Live

bit 7÷0 These bits are not used. For future compatibility write 0.

SnBSPS

Socket (receive, transmit) Buffer Start Pointer and Size register. Command code: 9. Size: 4 bytes. HC can write to this register only when the socket is disabled. User software must ensure that enabled sockets receive and transmit buffers do not overlap.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
31÷24	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	TBS<7:0>							
24÷16	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	TBSP<7:0>							
15÷8	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	RBS<7:0>							
7÷0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	RBSP<7:0>							

bit 31÷24 **TBS<7:0>:** Transmit Buffer Size register. Transmit buffer size = (TBS + 1) * 2048 - 1 bytes

bit 24÷16 **TBSP<7:0>:** Transmit Buffer Start Pointer register. Transmit buffer start pointer = TBSP * 2048

bit 15÷8 **RBS<7:0>:** Receive Buffer Size register. Receive buffer size = (RBS + 1) * 2048 - 1 bytes

bit 7÷0 **RBSP<7:0>:** Receive Buffer Start Pointer register. Receive buffer start pointer = RBSP * 2048

SnRDS

Socket Received Data Size register. Command code: 10. Size: 4 bytes. Read this register in order to find out how much data is in the receive buffer. This register value is valid only when the socket is enabled. When reading this register, the two LSB are clocked out first, followed by the the other two bytes (MSB). If the size of the receive buffer is less than 65536 bytes, HC could read only the first two bytes. This reduces overhead.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
31÷24	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RDS3<31:24>							
24÷16	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RDS2<23:16>							
15÷8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RDS1<15:8>							
7÷0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RDS0<7:0>							

bit 31÷24 **RDS3<31:24>:** Received Data Size byte 3

bit 23÷16 **RDS2<23:16>:** Received Data Size byte 2

bit 15÷8 **RDS1<15:8>:** Received Data Size byte 1

bit 7÷0 **RDS0<7:0>:** Received Data Size byte 0

SnTFS

Socket Transmit Free Size register. Command code: 11. Size: 4 bytes. Read this register in order to find out how much free space is in the transmit buffer. This register value is valid only when the socket is enabled. When reading this register, the two LSB are clocked out first, followed by the the other two bytes (MSB). If the size of the transmit buffer is less than 65536 bytes, HC could read only the first two bytes. This reduces overhead.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
31÷24	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFS3<31:24>							
24÷16	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFS2<23:16>							
15÷8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFS1<15:8>							
7÷0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFS0<7:0>							

bit 31÷24 **TFS3<31:24>:** Transmit Free Size byte 3

bit 23÷16 **TFS2<23:16>:** Transmit Free Size byte 2

bit 15÷8 **TFS1<15:8>:** Transmit Free Size byte 1

bit 7÷0 **TFS0<7:0>:** Transmit Free Size byte 0

SnMTU

Socket Maximum Transmission Unit. Command code: 12. Size: 2 bytes. HC can write this register with a value between 576 and 1500.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	R/W-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-1
	-	-	-	-	-	MTU<10:8>		
7÷0	R/W-1	R/W-1	R/W-0	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0
	MTU<7:0>							

bit 15÷11 These bits are not used. For future compatibility write 0.

bit 10÷0 **MTU<10:0>:** Maximum Transmission Unit. Default 1500.

SnPMSS

Socket Peer Maximum Segment Size. Command code: 12. Size: 2 bytes. This register is read only and follows register SnMTU. This register holds the MSS (maximum segment size) that the peer can accept. Socket logic writes this register after it receives and ACK packet to the SYN packet sent if in client mode, or after it receives a SYN packet when in server mode.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-1
	-	-	-	-	-	PMSS<10:8>		
7÷0	R/W-1	R/W-1	R/W-0	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0
	PMSS<7:0>							

bit 10÷0 **PMSS<10:0>:** Peer Maximum Segment Size.

SnTCPT

Socket TCP Tuning. Command code: 15. Size: 4 bytes. Care must be taken when altering this register. For most situations, the default values are the best choice.

byte offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
3	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
	ACKD<31:24>							
2	R/W-1	R/W-1	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
	RMIN<23:16>							
1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
	KPA<15:8>							
0	R-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-1	R/W-0	R/W-1
	TPF	SRTR<6:4>			DRTR<3:0>			

bit 31÷24 **ACKD<31:24>:** Acknowledgement Delay. The time unit is 5ms. This bit field only accepts values between 2 and 100.

bit 23÷16 **RMIN<23:16>:** Retransmission timeout lower limit. The time unit is 5ms. This bit field only accepts values higher than 1. The retransmission timeout upper limit is fixed at 60s.

bit 15÷8 **KPA<15:8>:** Keep Alive interval in units of 30s. A zero value means that the algorithm is disabled.

bit 7 **TPF:** Transmit Push Flag. When a TCP packet containing data is built, this bit value is written to the PUSH flag in the TCP header. HC can change this bit value when writes data to the transmit buffer over the SPI interface using command code 14. This bit is read only.

bit 6÷4 **SRTR<7:5>:** SYN packet Retransmission

bit 3÷0 **DRTR<7:5>:** Data packet Retransmission

SnCWSST

Socket Congestion Window and Slow Start Threshold register. Command code: 16. Size: 4 bytes.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
31÷24	R	R	R	R	R	R	R	R
	SST<15:8>							
23÷16	R	R	R	R	R	R	R	R
	SST<7:0>							
15÷8	R	R	R	R	R	R	R	R
	CW<15:8>							
7÷0	R	R	R	R	R	R	R	R
	CW<7:0>							

bit 31÷16 **SST<15:0>:** Slow Start Threshold register. This register is given in units of SnPMSS register.

bit 15÷0 **CW<15:0>:** Congestion Window. This register is given in units of SnPMSS register.

SnRTO

Socket Retransmit Timeout. Command code: 17. Size: 2 bytes. This register is given in units of 5ms. It represents the period of time the retransmit timeout timer is loaded (if it is not already running) when a packet is sent out.

bit range	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
15÷8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RTO1<15:8>							
7÷0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RTO0<7:0>							

bit 15÷8 **RTO1<15:8>:** Retransmit timeout byte 1

bit 7÷0 **RTO0<7:0>:** Retransmit timeout byte 0

SnRHP

Socket Receive buffer Head Pointer. Command code: 28. Size: 4 bytes. This register is read only. When reading this register, two LSB are clocked out first followed by the remaining two bytes. This is to reduce overhead when the receive buffer size is less than 65536 bytes. This register is provided for debug purposes. It may be removed in the future releases. User production software should not access this register.

SnRTP

Socket Receive buffer Tail Pointer. Command code: 29. Size: 4 bytes. This register is read only. When reading this register, two LSB are clocked out first followed by the remaining two bytes. This is to reduce overhead when the receive buffer size is less than 65536 bytes. This register is provided for debug purposes. It may be removed in the future releases. User production software should not access this register.

SnTHP

Socket Transmit buffer Head Pointer. Command code: 30. Size: 4 bytes. This register is read only. When reading this register, two LSB are clocked out first followed by the remaining two bytes. This is to reduce overhead when the receive buffer size is less than 65536 bytes. This register is provided for debug purposes. It may be removed in the future releases. User production software should not access this register.

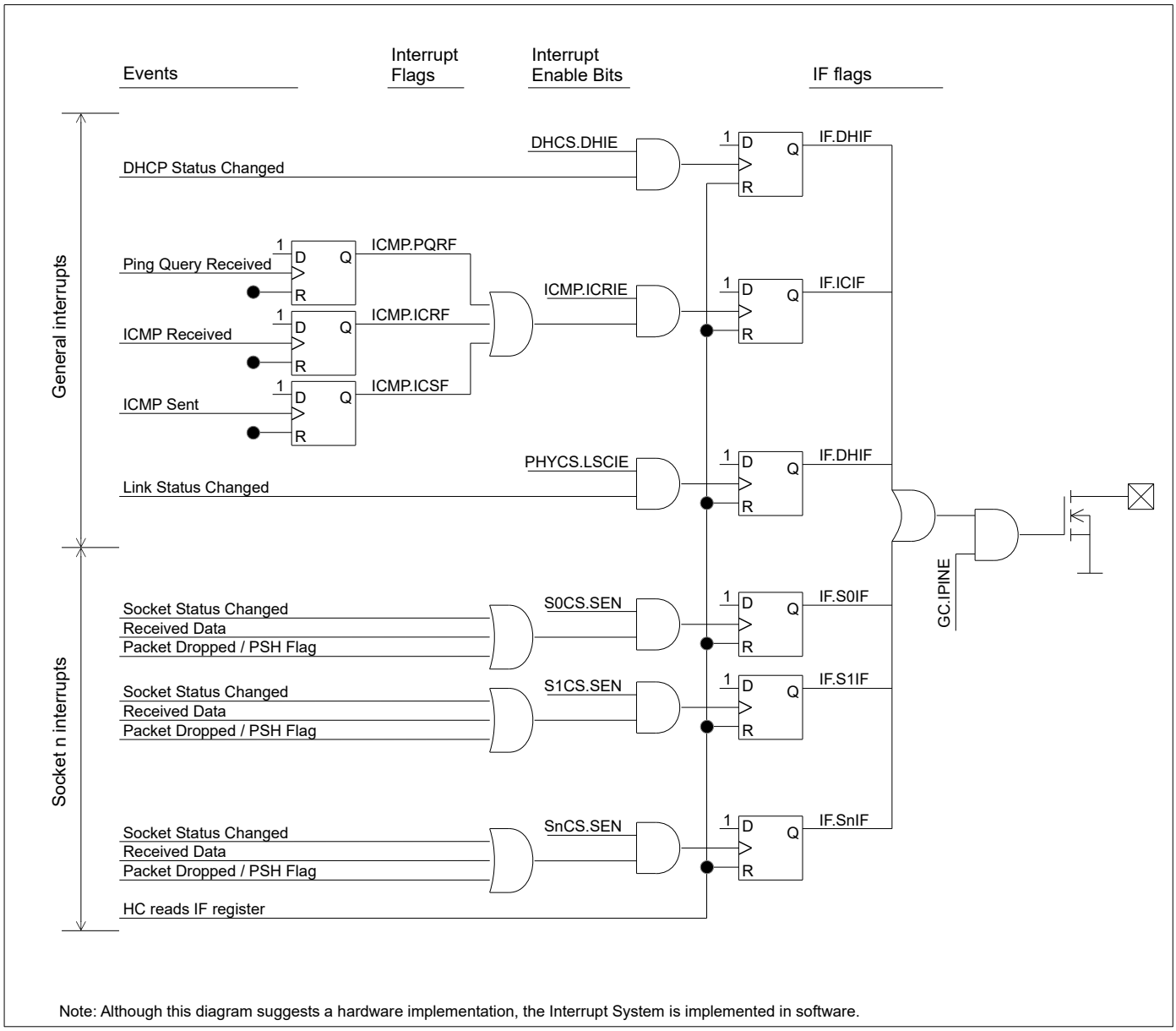
SnTTP

Socket Transmit buffer Tail Pointer. Command code: 31. Size: 4 bytes. This register is read only. When reading this register, two LSB are clocked out first followed by the remaining two bytes. This is to reduce overhead when the receive buffer size is less than 65536 bytes. This register is provided for debug purposes. It may be removed in the future releases. User production software should not access this register.

13 Interrupt System

The interrupt system provides status notifications to HC for events that are difficult to predict if and when they occur. It is an alternative to constantly polling various registers that would indicate that.

Fig. 13-1: Interrupt system block diagram



Note: Although this diagram suggests a hardware implementation, the Interrupt System is implemented in software.

All interrupts are routed to a single bit IPINS (Interrupt PIN Status) in the GS (General Status) register. This bit can be mirrored to a physical pin which if connected, HC can read or trigger an interrupt.

By default, bit IPINS is cleared, which means that the pin is configured as input and no interrupt can be issued. When HC sets bit IPINE in GC register the pin becomes an open drain output. To have the pin in logic high, an external pull-up resistor is needed. This pin is 5V tolerant. Bit IPINS, and the pin can be cleared by reading register IF.

SEI sends out the General Status register value with every SPI transaction. This helps in case the physical interrupt pin is not used.

If DHCP interrupts are enabled by having bit DHIE in DHCS register set, an interrupt is issued when DHCP obtains an IP address from a DHCP server, when it louses it or if DHCP fails to obtain an IP address. When an interrupt is issued, bit DHIF in IF register is set.

Interrupt flags PQRF, ICRF and ICSF in ICMP register are or-ed to bit ICIF in IF register. To enable these interrupts, set the associated interrupt enable bit in ICMP register.

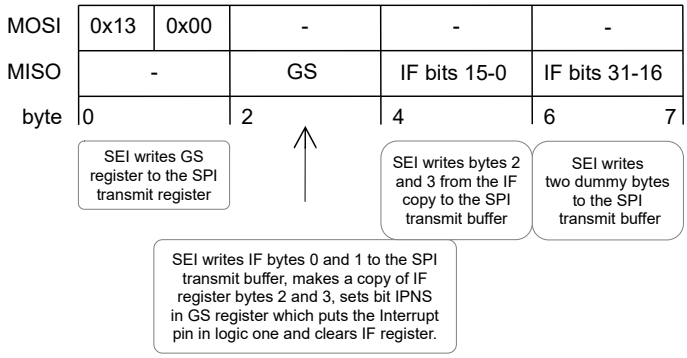
LSCIF bit is set when PHY Link status changes. To enable this interrupt, set bit LSCIE in PHYCS register.

Each socket has an interrupt flag in IF register. A socket issues an interrupt when:

– its status changes
– it receives a UDP packet
– in UDP mode and it drops a received packet because the receive buffer is full or when in TCP mode it receives a packet with the PUSH flag set

There is no mechanism to individually clear interrupt flags in IF register. HC can keep a local copy of the IF register and work with that copy. HC then handle one event at a time and clear the respective bit in the local copy. When reading the IF register from the SPI interface, HC then OR the local copy with the one just read.

Fig. 13-2: Read IF register example



14 Register Monitor Interface

The Register Monitor Interface (RMI) is made up of a software routine, a UART peripheral and the Tx pin. RMI continuously outputs the value of almost all registers. RMI can be enabled / disabled by writing a one / zero to bit RMIEN in GC (General Control register). RMI is enabled by default and starts one second after reset. When not used, the Tx pin should be left unconnected even if the RMI is not enabled (UART Tx is always enabled). The baud rate is fixed at 115200bps, one stop bit, no parity bit.

RMI sends the value of one or more registers wrapped in a tiny packet. A packet is made up of a header and payload data. Up to eight bytes are sent continuously every 5ms. Some longer packets are sent in chunks, each at 5ms interval.

The registers are split in two categories: common registers and socket registers.

For common registers the packet header is made up of one byte, which represents the packet ID and it is an even number starting from zero.

For socket registers the packet header is made up of two bytes, one byte which represents the packet ID which is an odd number starting from one, and a second byte which holds the socket number the register belongs.

Numeric values are sent in little-endian order with byte 0 indicated in the following table as b0, followed by byte 1 as b1 and so on. A MAC and IP address, from left to right, starts with byte 0 indicated as b0 and continues with byte 1 as b1 and so on.

Table 14-1: RMI packets for common registers.

Id	Register	Example
0	GC, GS	0x00, b0, b1, b0, b1
2	HMA	0x02, b1, b2, b3, b4, b5, b6
4	HIPA, HSM, GIPA	0x04, b2, b1 , b4 , b3, b2, b1, b4, b3, b2, b1, b4, b3
6	IF	0x08, b0, b1, b2, b3
8	PHYCS, ICMP	0x0A, b0, b1, b0, b1
10	PQIP	0x0C, b2, b1 , b4 , b3
12	DHCS	0x0E, b0, b1
14	DHSRV, DNS1, DNS2	0x10, b2, b1 , b4 , b3, b2, b1, b4, b3, b2, b1, b4, b3
16	DHRLT, DHLT	0x12, b0, b1, b2, b3, b0, b1, b2, b3
18	TRISB, PORTB, ODCB	0x14, b0, b1, b0, b1, b0, b1
20	TRISE, PORTE, ODCE	0x16, b0, b0, b0
22	ARPC, ATTL	0x18, b0, b1, b0, b1
24	FW	0x1C, b0, b1, b2, b3

Although the ARP table is not accessible via the SPI interface, it is sent out as part of socket registers. Row 0 is sent with socket 0 registers, row 1 with socket 1 registers and so on.

Table 14-2: RMI packets for ARP and socket registers.

Id	Register	Example
1	ARP Status, IP address	0x01, b0, b1, b2, b1, b4, b3
3	ARP MAC address	0x03, b1, b2, b3, b4, b5, b6
5	SnCS	0x05, b0, b1. Bits 12 and 13 (bits DRB and TBE) are always sent as zero. User can figure out their values by looking at Rx and/or Tx buffer pointers instead.
7	SnIPA	0x07, b2, b1, b4, b3
9	SnSM	0x09, b2, b1, b4, b3
11	SnPIPA	0x0B, b2, b1, b4, b3
13	SnP, SnPP	0x0D, b0, b1, b0, b1
15	SnBSPS	0x0F, b0, b1, b2, b3
17	SnTTL, SnMTU, SnPMSS	0x11, b0, b0, b1, b0, b1
19	SnRTO, SnSRTT	0x13, b0, b1, b0, b1
21	SnCW, SnSST	0x15, b0, b1, b0, b1
23	SnRHP	0x17, b0, b1, b2, b3
25	SnRTP	0x19, b0, b1, b2, b3
27	SnTHP	0x1B, b0, b1, b2, b3
29	SnTHHP	0x1D, b0, b1, b2, b3
31	SnTTP	0x1F, b0, b1, b2, b3

15 ARSHXIP vs ARSHXNS vs ARSHZIP

Table 14-1: Differences between ARSHXIP, ARSHXNS and ARSHZIP

	ARSHXIP	ARSHXNS	ARSHZIP
Short description	Ethernet Shield with a low cost PIC32 microcontroller	Ethernet Shield with two Ethernet ports	Ethernet Shield with encryption
Microcontroller	PIC32MX664F064H	PIC32MX695F512H	PIC32MZ0512EFK064
Ethernet ports	1	2	1
PHY device, network switch	IP101GR controlled via registers PHYCS, PHYA and PHYD	KSZ8863RLL controlled via registers NSC and NSRAD	IP101GR controlled via registers PHYCS, PHYA and PHYD
Number of sockets	4	16	16
Buffers memory total	20480 bytes	106496 bytes	106496 bytes
GPIO	15	14	14
Encryption	No	No	Yes

End of this document.