Clause 22 Access to Clause 45 Registers

802.3ah - November 2002

Don Pannell
Marvell Semiconductor
dpannell@marvell.com

Supporters of this Proposal

- Ed Turner, Lattice Semiconductor
- David Law, 3 Com
- Scott Simon, Cisco
- Hugh Barrass, Cisco
- Matt Squire, Hatteras Networks
- Kevin Daines, World Wide Packets
- Ulf Jonsson, Ericsson
- Ben Brown, AMCC
- Bradley Booth, Intel
- Vipul Bhatt

I’d like to thank all of these people for help with reviews and solutions to problems

Overview

- Clause 45 defined a new register access method with a larger address space
- Clause 45 was 1st used for new 10 Gig PHYs and MACs (802.3ae)
- Since both 10 Gig PHYs and MACs were new designs this approach worked well
- Clause 45 appeared to solve 802.3's register space problem forever (It didn't)

The Problem

- Most 802.3ah PHYs need to use the larger address space defined by Clause 45
- Most 802.3ah PHYs want to work with existing 10/100 MACs using MII for frame data & MDC/MDIO for register access
- Most Existing 10/100 MACs can’t do Clause 45! They can only do Clause 22!
- “Houston, we have a problem!”

The Solution

- We need to define a standard way to access Clause 45 registers using Clause 22
- Using a standard ‘backwards compatible’ way to access Clause 45 registers WILL solve 802.3’s register access problems for 802.3ah and beyond
- This must be defined NOW since there are only 2 unused Clause 22 registers left

The Implementation

- Use Clause 22 Register 13 as a Clause 45 Command register

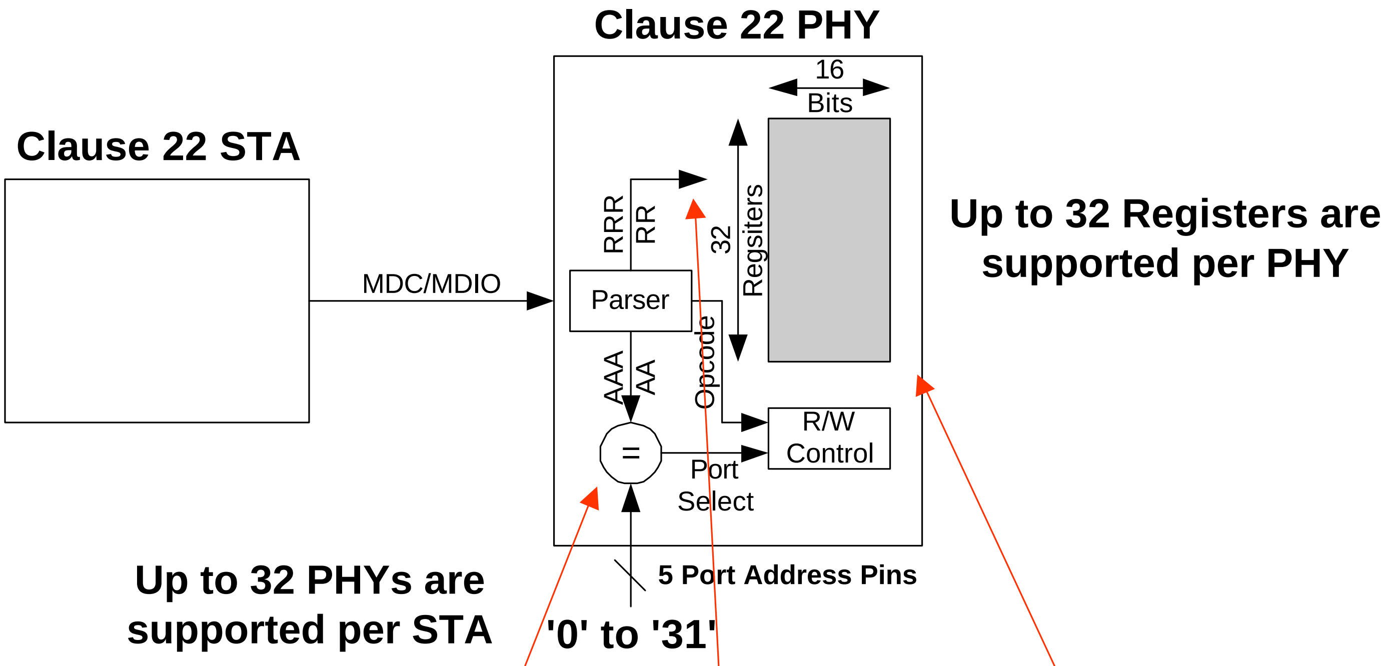
- Use Clause 22 Register 14 as a Clause 45 Address/Data register

Clause 22 vs. Clause 45

- Clause 22:
  - 2 Opcodes
    - Read & Write
  - 32 Ports
  - 32 Registers per Port

- Clause 45:
  - 4 Opcodes
    - Address, Read, Write & Read Increment
  - 32 Ports
  - 32 Devices per Port
  - 64K Registers per Device

Clause 22 STA & PHY



Up to 32 Registers are supported per PHY

Up to 32 PHYs are supported per STA

	Management Frame Fields - Clause 22							
	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
Read	1 . . .1	01	10	AAAAA	RRRRR	Z0	DDDDDDDDDDDDDDDD	Z
Write	1 . . .1	01	01	AAAAA	RRRRR	10	DDDDDDDDDDDDDDDD	Z

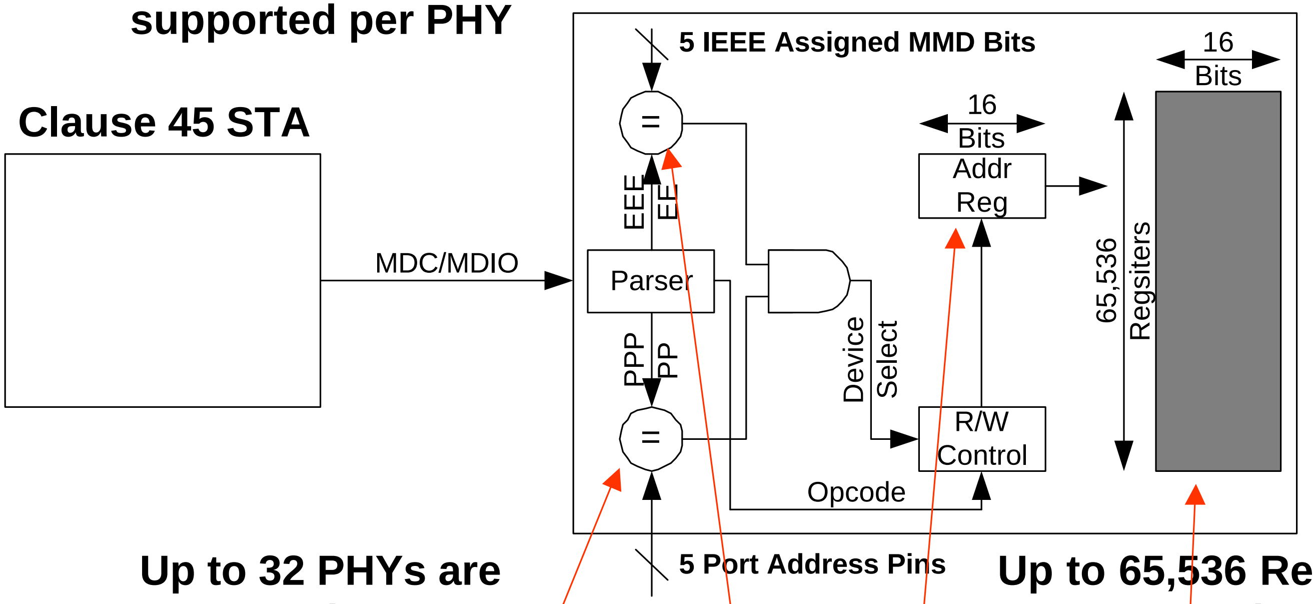
Operation of Clause 22

- To Read a Clause 22 Register Perform:
  - Read Register RRRRR from PHY AAAAA
- To Write a Clause 22 Register Perform:
  - Write Register RRRRR to PHY AAAAA
- Each Operation Takes 1 Step

Clause 45 STA & PHY

Up to 32 MMDs supported per PHY

Clause 45 PHY



Clause 45 STA

MDC/MDIO

5 IEEE Assigned MMD Bits

16 Bits

=

EEE

EE

Parser

PPP

PP

=

Device Select

16 Bits

Addr Reg

R/W Control

Opcode

65,536 Regsiters

5 Port Address Pins

Up to 32 PHYs are supported per STA

'0' to '31'

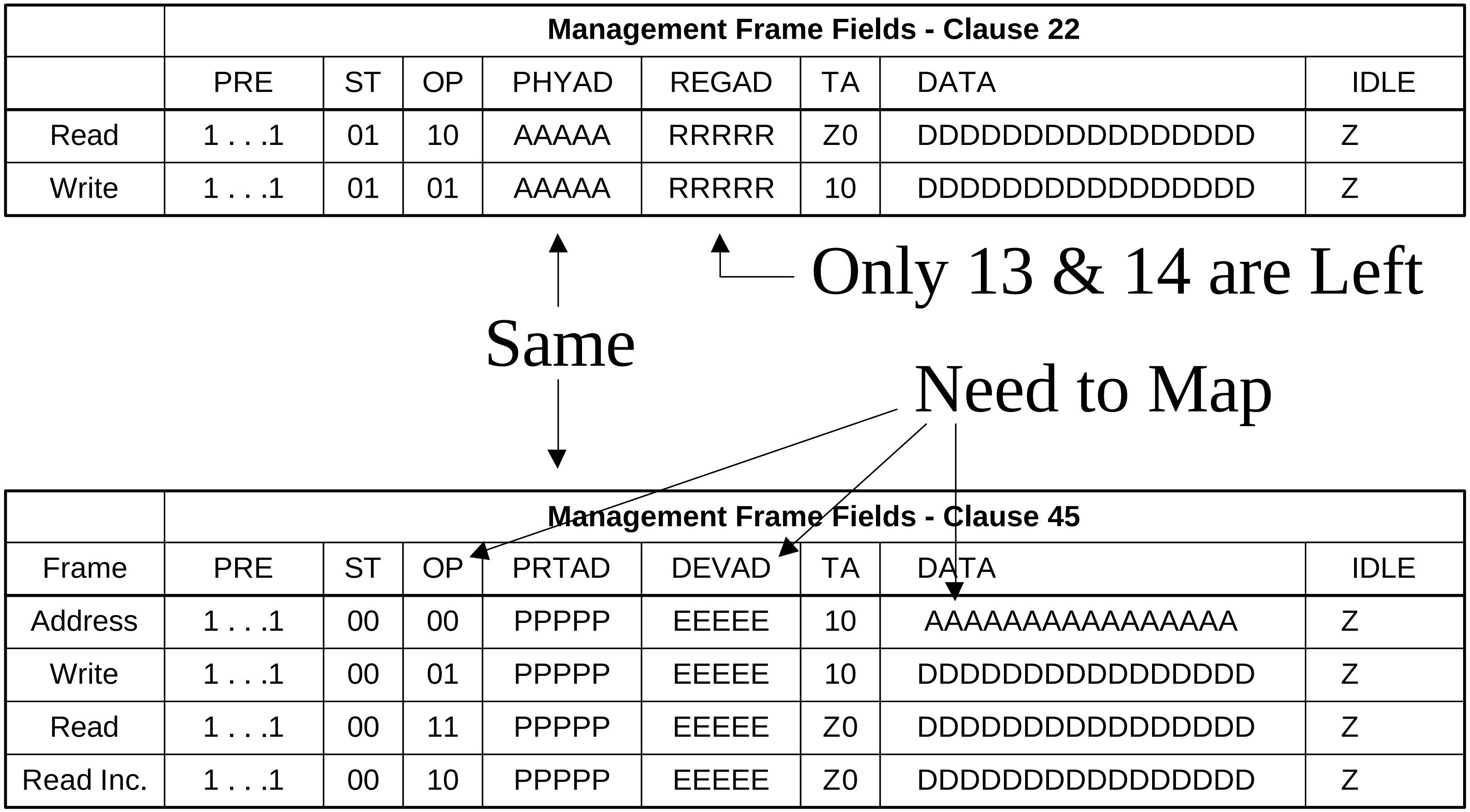
Up to 65,536 Regsters are supported per MMD

	Management Frame Fields - Clause 45							
Frame	PRE	ST	OP	PRTAD	DEVAD	TA	DATA	IDLE
Address	1 . . .1	00	00	PPPPP	EEEEE	10	AAAAAAAAAAAAAAAA	Z
Write	1 . . .1	00	01	PPPPP	EEEEE	10	DDDDDDDDDDDDDDDD	Z
Read	1 . . .1	00	11	PPPPP	EEEEE	Z0	DDDDDDDDDDDDDDDD	Z
Read Inc.	1 . . .1	00	10	PPPPP	EEEEE	Z0	DDDDDDDDDDDDDDDD	Z

Operation of Clause 45

- To Read a Clause 45 Register Perform:
  - Write Address AAAAAAAAAAAAAAAA to Device EEEEE on Port PPPPP
  - Read Register From Device EEEEE on Port PPPPP
- To Write a Clause 45 Register Perform:
  - Write Address AAAAAAAAAAAAAAAA to Device EEEEE on Port PPPPP
  - Write Register To Device EEEEE on Port PPPPP
- Each Operation Takes 2 Steps

Clause 22 vs. Clause 45



	Management Frame Fields - Clause 22							
	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
Read	1 . . .1	01	10	AAAAA	RRRRR	Z0	DDDDDDDDDDDDDDDD	Z
Write	1 . . .1	01	01	AAAAA	RRRRR	10	DDDDDDDDDDDDDDDD	Z

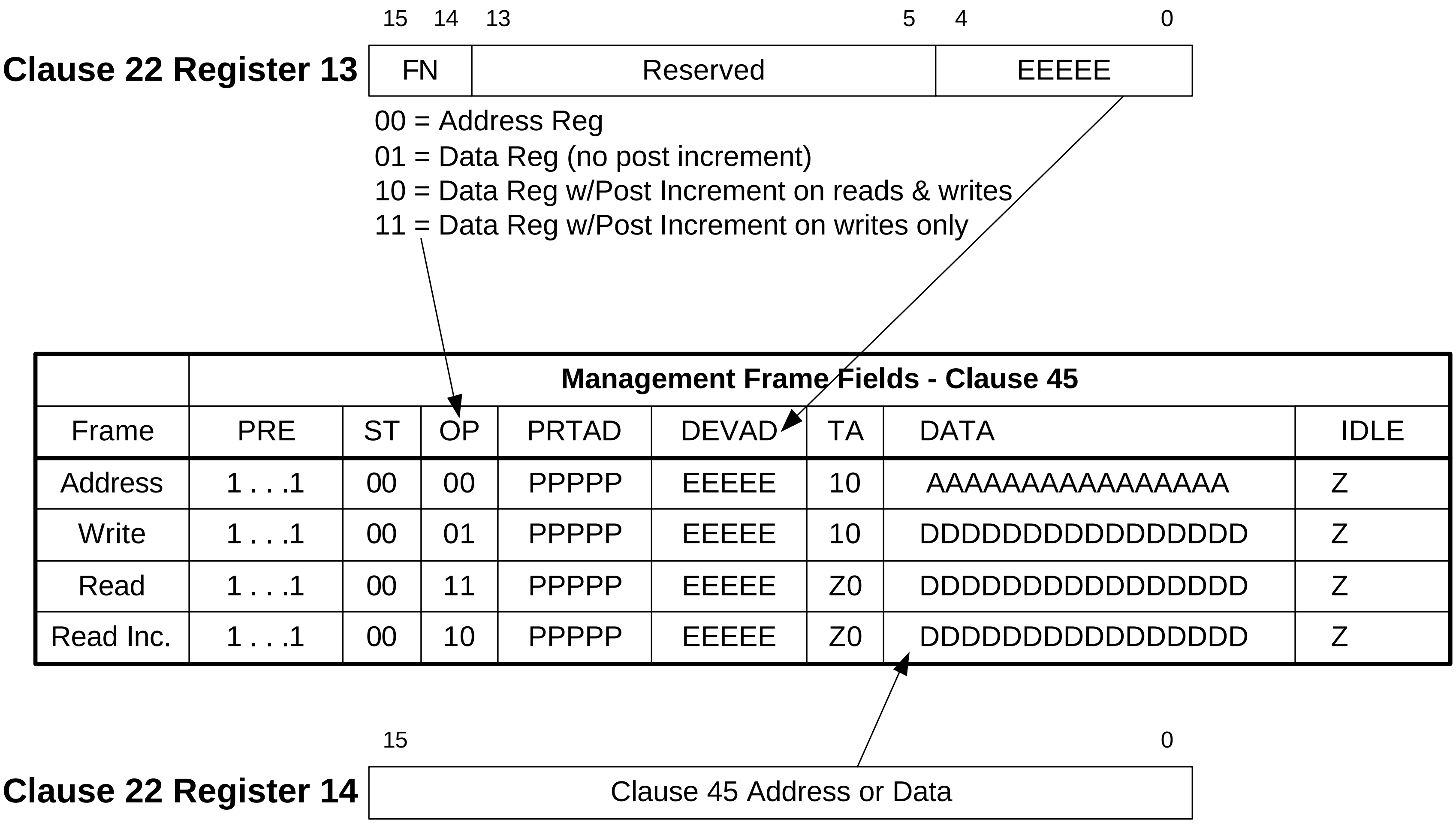
Only 13 & 14 are Left

Same

Need to Map

	Management Frame Fields - Clause 45							
Frame	PRE	ST	OP	PRTAD	DEVAD	TA	DATA	IDLE
Address	1 . . .1	00	00	PPPPP	EEEEE	10	AAAAAAAAAAAAAAAA	Z
Write	1 . . .1	00	01	PPPPP	EEEEE	10	DDDDDDDDDDDDDDDD	Z
Read	1 . . .1	00	11	PPPPP	EEEEE	Z0	DDDDDDDDDDDDDDDD	Z
Read Inc.	1 . . .1	00	10	PPPPP	EEEEE	Z0	DDDDDDDDDDDDDDDD	Z

Mapping Using Reg 13 & 14



Clause 22 Register 13

15 14	13 5	4 0
FN	Reserved	EEEEE

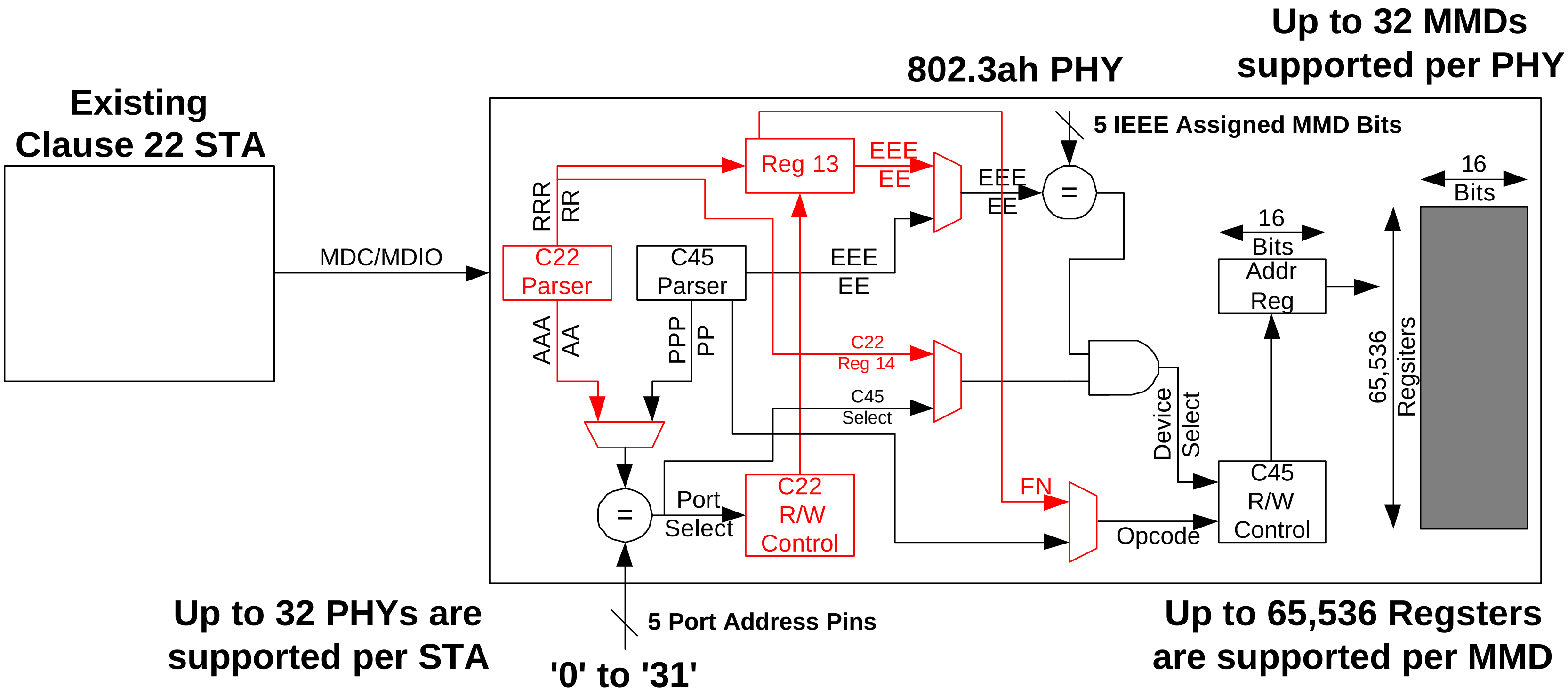
00 = Address Reg
01 = Data Reg (no post increment)
10 = Data Reg w/Post Increment on reads & writes
11 = Data Reg w/Post Increment on writes only

	Management Frame Fields - Clause 45							
Frame	PRE	ST	OP	PRTAD	DEVAD	TA	DATA	IDLE
Address	1 . . .1	00	00	PPPPP	EEEEE	10	AAAAAAAAAAAAAAAA	Z
Write	1 . . .1	00	01	PPPPP	EEEEE	10	DDDDDDDDDDDDDDDD	Z
Read	1 . . .1	00	11	PPPPP	EEEEE	Z0	DDDDDDDDDDDDDDDD	Z
Read Inc.	1 . . .1	00	10	PPPPP	EEEEE	Z0	DDDDDDDDDDDDDDDD	Z

Clause 22 Register 14

15 0
Clause 45 Address or Data

Clause 22 STA w/.ah PHY



Existing Clause 22 STA

802.3ah PHY

Up to 32 MMDs supported per PHY

Up to 32 PHYs are supported per STA

Up to 65,536 Regsters are supported per MMD

- Clause 22 Logic added to Clause 45 PHY is shown in RED

Clause 22 STA w/.ah PHY

- 802.3ah PHYs Respond to both Clause 22 Frames and Clause 45 Frames
- If Frame is Clause 22 the MUX'es select the RED signals
- If Frame is Clause 45 the MUX'es select the BLACK signals

Operation of C22 to C45

- To Read a C45 Register C22 Perform:
  - Write FN = Address & EEEEE to C22 Register 13 on Port PPPPP
  - Write Address AAAAAAAAAAAAAAAA to C22 Register 14 on Port PPPPP
  - Write FN = Data & EEEEE to C22 Register 13 on Port PPPPP
  - Read Register From C22 Register 14 on Port PPPPP
- Read Operation Takes 4 Steps

Operation of C22 to C45

- To Write a C45 Register C22 Perform:
  - Write FN = Address & EEEEE to C22 Register 13 on Port PPPPP
  - Write Address AAAAAAAAAAAAAAAA to C22 Register 14 on Port PPPPP
  - Write FN = Data & EEEEE to C22 Register 13 on Port PPPPP
  - Write Register To C22 Register 14 on Port PPPPP
- Only the Last Step is Different from Read

C45 Set Address using C22

- Write 00xx..xxEEEEE to Clause 22 Reg 13
- Write AAAAAAAAAAAAAAAA to Reg 14

- Subsequent writes to Reg 14 continue to re-write Device EEEEE's address register until Reg 13 is modified
- Subsequent reads from Reg 14 return Device EEEEE's current address register until Reg 13 is modified

C45 Write Data using C22

- Write 01xx..xxEEEEE to Clause 22 Reg 13
- Write DDDDDDDDDDDD to Reg 14

- Subsequent writes to Reg 14 continue to re-write Device EEEEE's data register pointed to by the last Set Address until Reg 13 is modified
- Subsequent reads from Reg 14 return Device EEEEE's current data register pointed to by the last Set Address until Reg 13 is modified

C45 Read Data using C22

- Write 01xx..xxEEEEE to Clause 22 Reg 13
- Read DDDDDDDDDDDD from Reg 14

- Subsequent reads from Reg 14 continue to re-read Device EEEEE's data register pointed to by the last Set Address until Reg 13 is modified
- Subsequent writes to Reg 14 write Device EEEEE's current data register pointed to by the last Set Address until Reg 13 is modified

C45 Read Inc. using C22

- Write 10xx..xxEEEEE to Clause 22 Reg 13
- Read DDDDDDDDDDDD from Reg 14
- Read DDDDDDDDDDDD from Reg 14 . . .

- Subsequent reads from Reg 14 read Device EEEEE’s next higher addressed data register (using post increment) until Reg 13 is modified
- Subsequent writes to Reg 14 write to Device EEEEE’s next higher addressed data register (using post increment) until Reg 13 is modified

New Write Inc. using C22

- Write 10xx..xxEEEEE to Clause 22 Reg 13
- Write DDDDDDDDDDDD to Reg 14
- Write DDDDDDDDDDDD to Reg 14 . . .

- Subsequent writes to Reg 14 write Device EEEEE's next higher addressed data register (using post increment) until Reg 13 is modified
- Subsequent reads from Reg 14 read Device EEEEE's next higher addressed data register (using post increment) until Reg 13 is modified

New RMW Inc. using C22

- Write 11xx..xxEEEEE to Clause 22 Reg 13
- Read DDDDDDDDDDDD from Reg 14
- Write DDDDDDDDDDDD to Reg 14 . . .

- Subsequent writes to Reg 14 write Device EEEEE's next higher addressed data register (using post increment) until Reg 13 is modified
- Subsequent reads from Reg 14 read Device EEEEE's next higher addressed data register (NO post increment) until Reg 13 is modified

Reads to Reg 13 using C22

- A copy of Clause 22 Reg 13 exists in all MMDs in a Port (a write updates all copies)
- When a Read to Clause 22 Reg 13 occurs which MMD responds? (It doesn’t matter which one since they are all the same)
- The MMD currently selected by Clause 22 Reg 13’s EEEEE bits (the DEVAD bits) is the only MMD to responds to Reg 13 Reads

Benefits

- All Ports, Devices and Registers supported in Clause 45 are accessible to Clause 22 MDC/MDIO STA (station management entity) devices (typically MACs)
- Clause 22 only devices can co-exist with Clause 45 devices that support this proposal (as long as they use Unique Port Addresses)

Who Does this Effect?

- Existing Clause 22 devices do not need to be modified (a major goal of this proposal)
- No modification of 802.3ae (10 Gig) Clause 45 devices
- New 802.3ah (EFM) Clause 45 PHY devices will need to work with Clause 45 MDC/MDIO STAs AND work with Clause 22 MDC/MDIO STAs using Registers 13 & 14

Summary

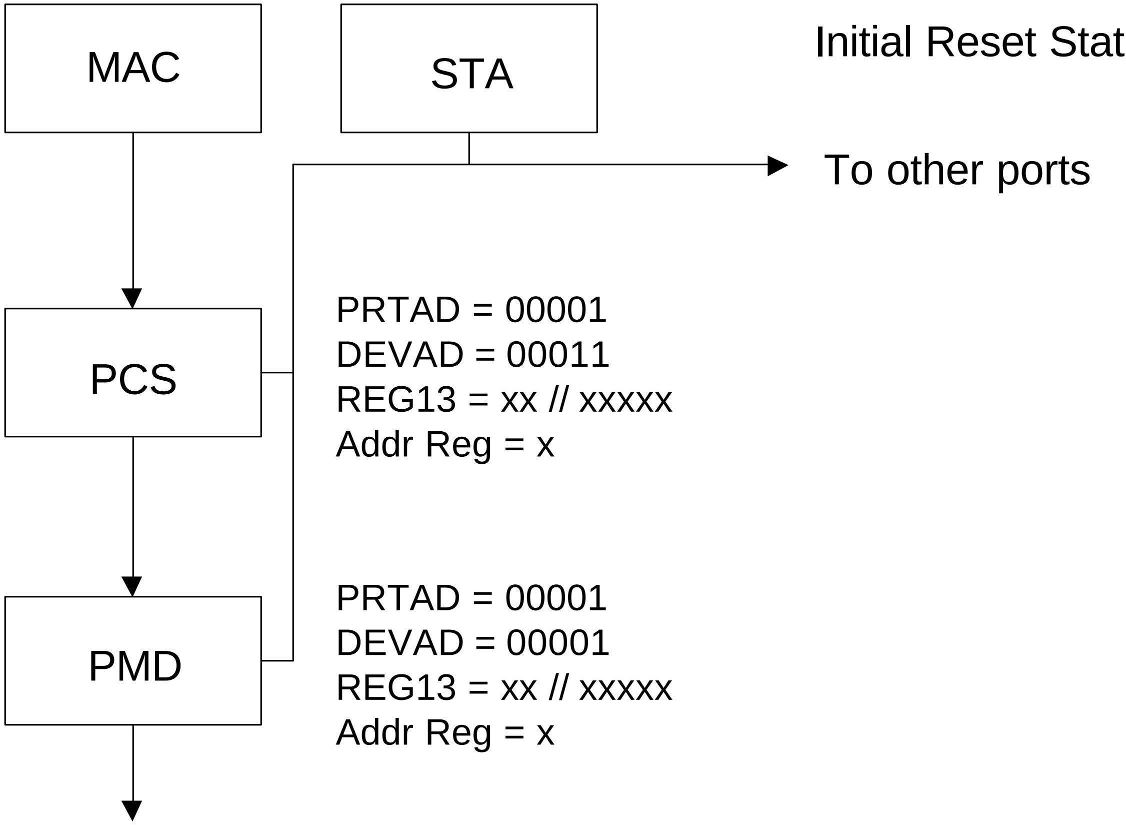
- This is our last chance to allow older CPUs and MACs to work with newer 802.3 PHYs
  - We are running out of Clause 22 Registers
- The MII data path is only 1/2 of the compatibility problem
  - Clause 22 MDC/MDIO STAs must work too!
- The Electrical Interface Levels in Clause 45 needs to be modified (to 3.3V tolerant)
- Thanks

Appendix

- Example of this proposal in action
- Port Address Issue
- Register 13 Opcode options - why they are what they are

Example - Part 1

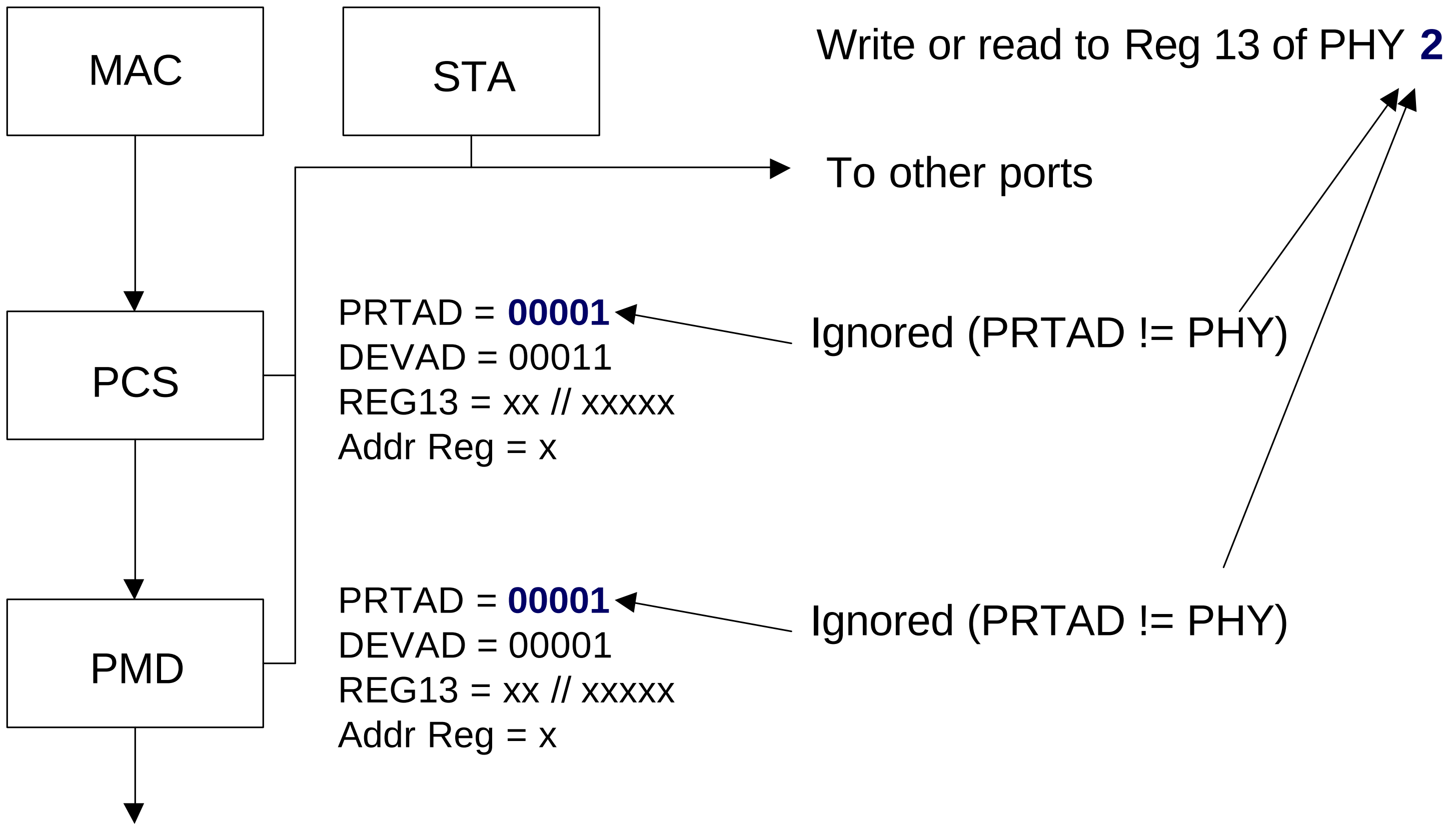
Initial Reset States



PRTAD = 00001
DEVAD = 00011
REG13 = xx // xxxxx
Addr Reg = x

PRTAD = 00001
DEVAD = 00001
REG13 = xx // xxxxx
Addr Reg = x

Example - Part 2



Write or read to Reg 13 of PHY **2**

To other ports

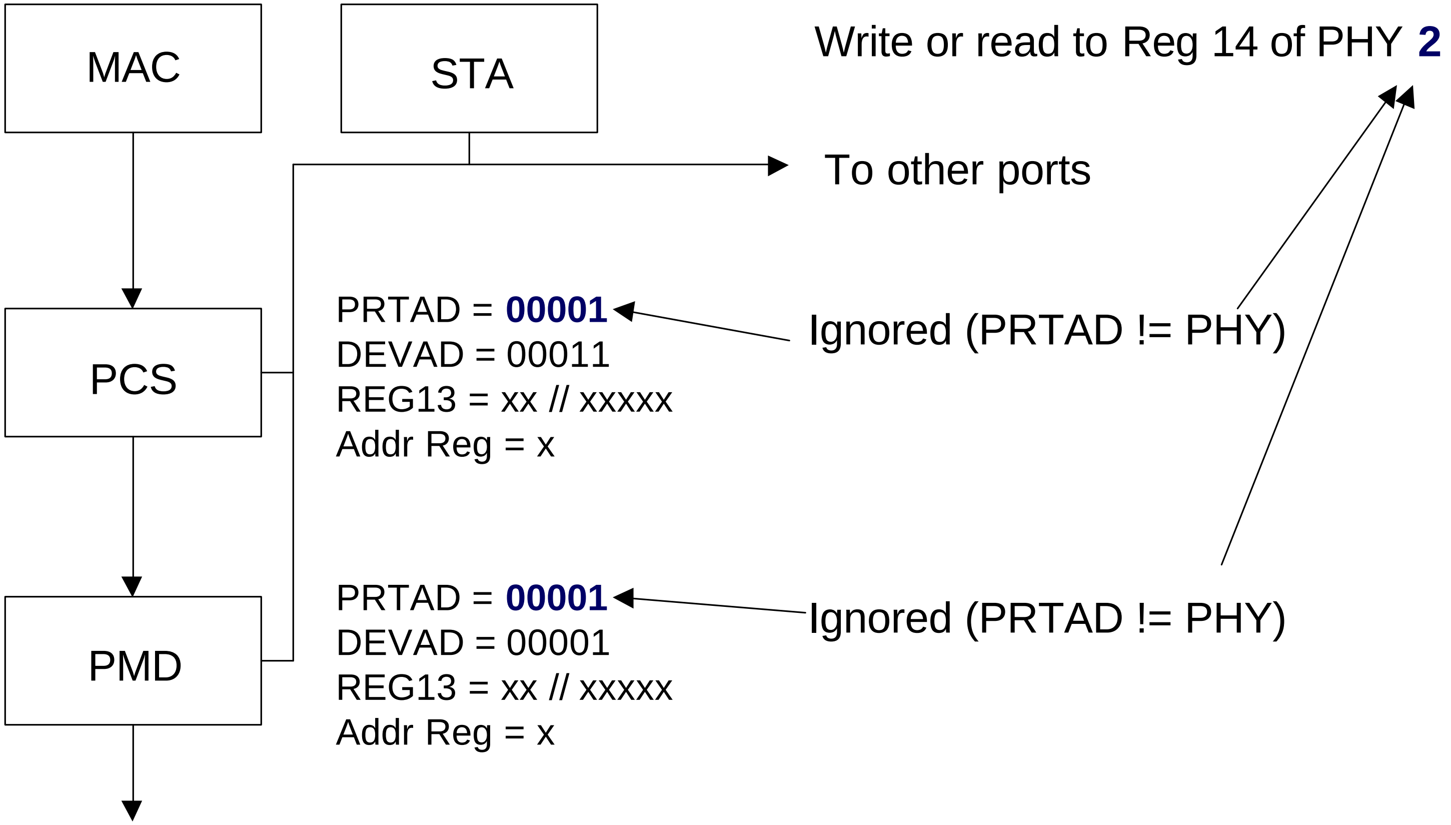
PRTAD = **00001**
DEVAD = 00011
REG13 = xx // xxxxx
Addr Reg = x

Ignored (PRTAD != PHY)

PRTAD = **00001**
DEVAD = 00001
REG13 = xx // xxxxx
Addr Reg = x

Ignored (PRTAD != PHY)

Example - Part 3



MAC

STA

Write or read to Reg 14 of PHY **2**

To other ports

PCS

PRTAD = **00001**
DEVAD = 00011
REG13 = xx // xxxxx
Addr Reg = x

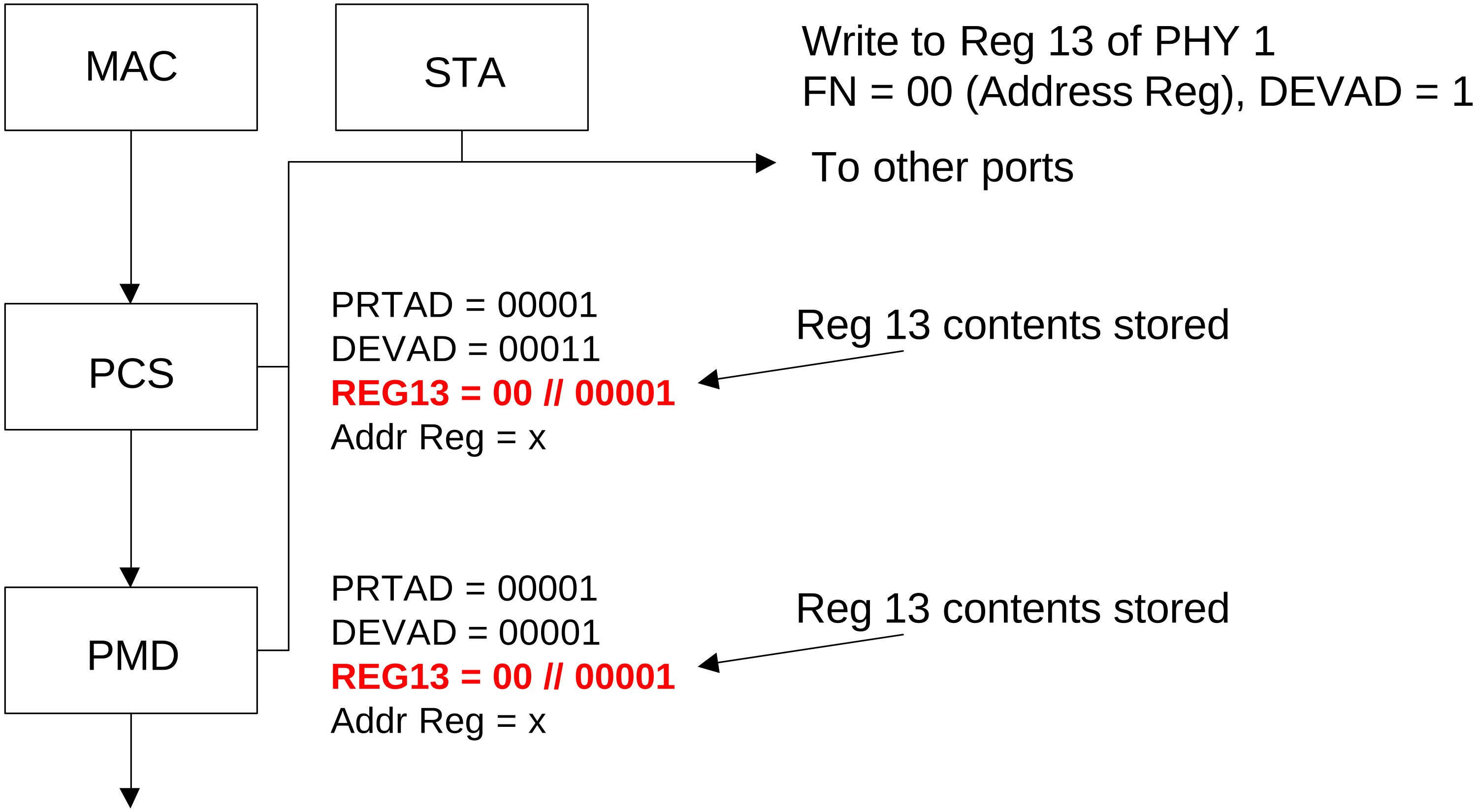
Ignored (PRTAD != PHY)

PMD

PRTAD = **00001**
DEVAD = 00001
REG13 = xx // xxxxx
Addr Reg = x

Ignored (PRTAD != PHY)

Example - Part 4



Write to Reg 13 of PHY 1
FN = 00 (Address Reg), DEVAD = 1

To other ports

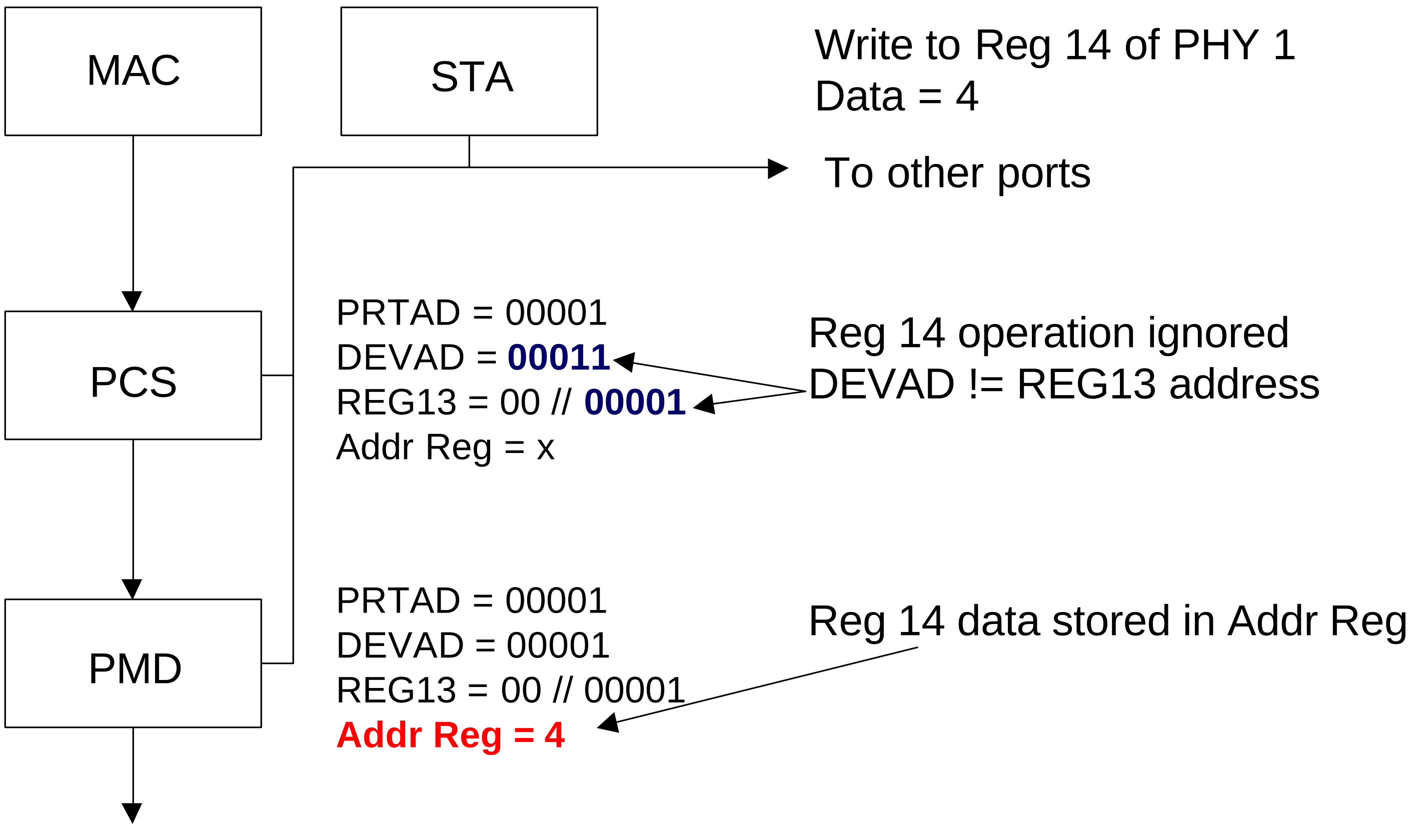
PRTAD = 00001
DEVAD = 00011
REG13 = 00 // 00001
Addr Reg = x

Reg 13 contents stored

PRTAD = 00001
DEVAD = 00001
REG13 = 00 // 00001
Addr Reg = x

Reg 13 contents stored

Example - Part 5



MAC

STA

Write to Reg 14 of PHY 1
Data = 4

To other ports

PCS

PRTAD = 00001
DEVAD = **00011**
REG13 = 00 // **00001**
Addr Reg = x

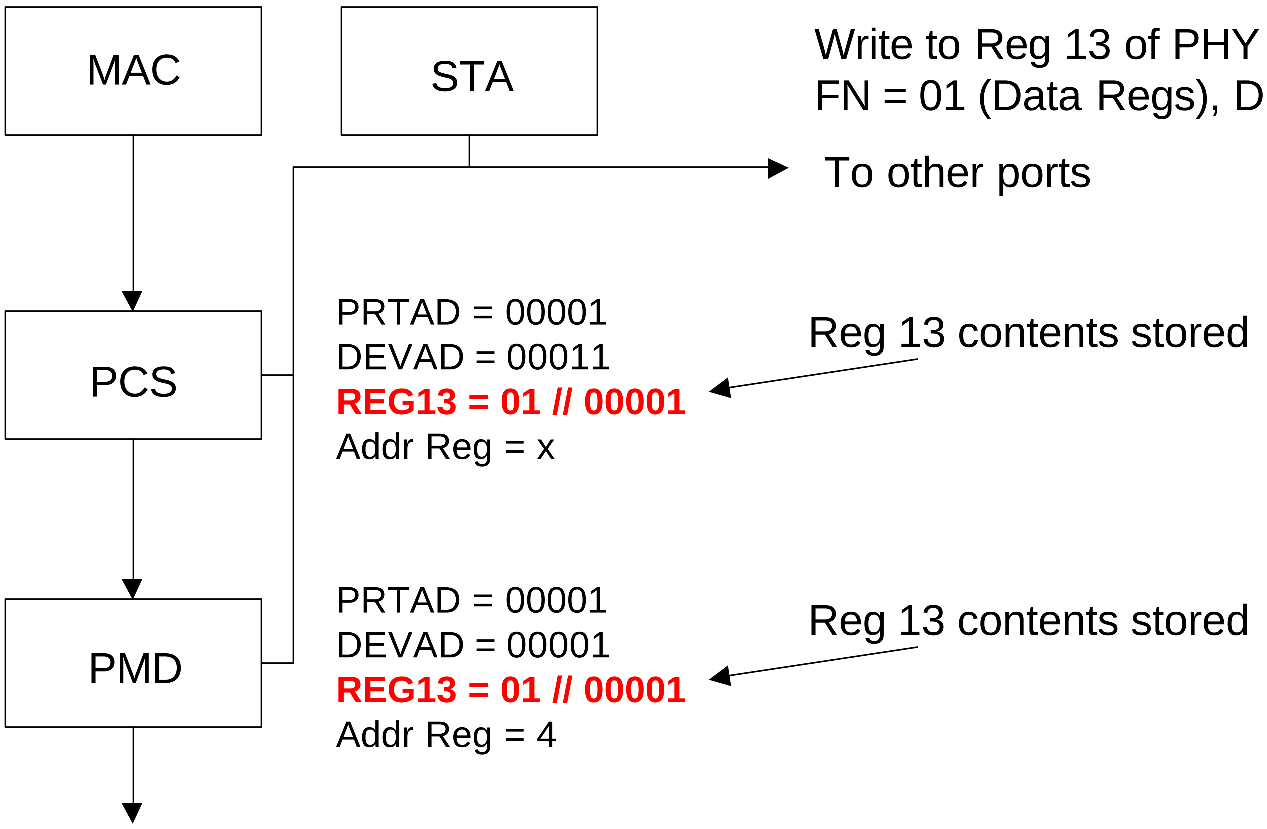
Reg 14 operation ignored
DEVAD != REG13 address

PMD

PRTAD = 00001
DEVAD = 00001
REG13 = 00 // 00001
Addr Reg = 4

Reg 14 data stored in Addr Reg

Example - Part 6



MAC

STA

Write to Reg 13 of PHY 1
FN = 01 (Data Regs), DEVAD = 1

To other ports

PCS

PRTAD = 00001
DEVAD = 00011
REG13 = 01 // 00001
Addr Reg = x

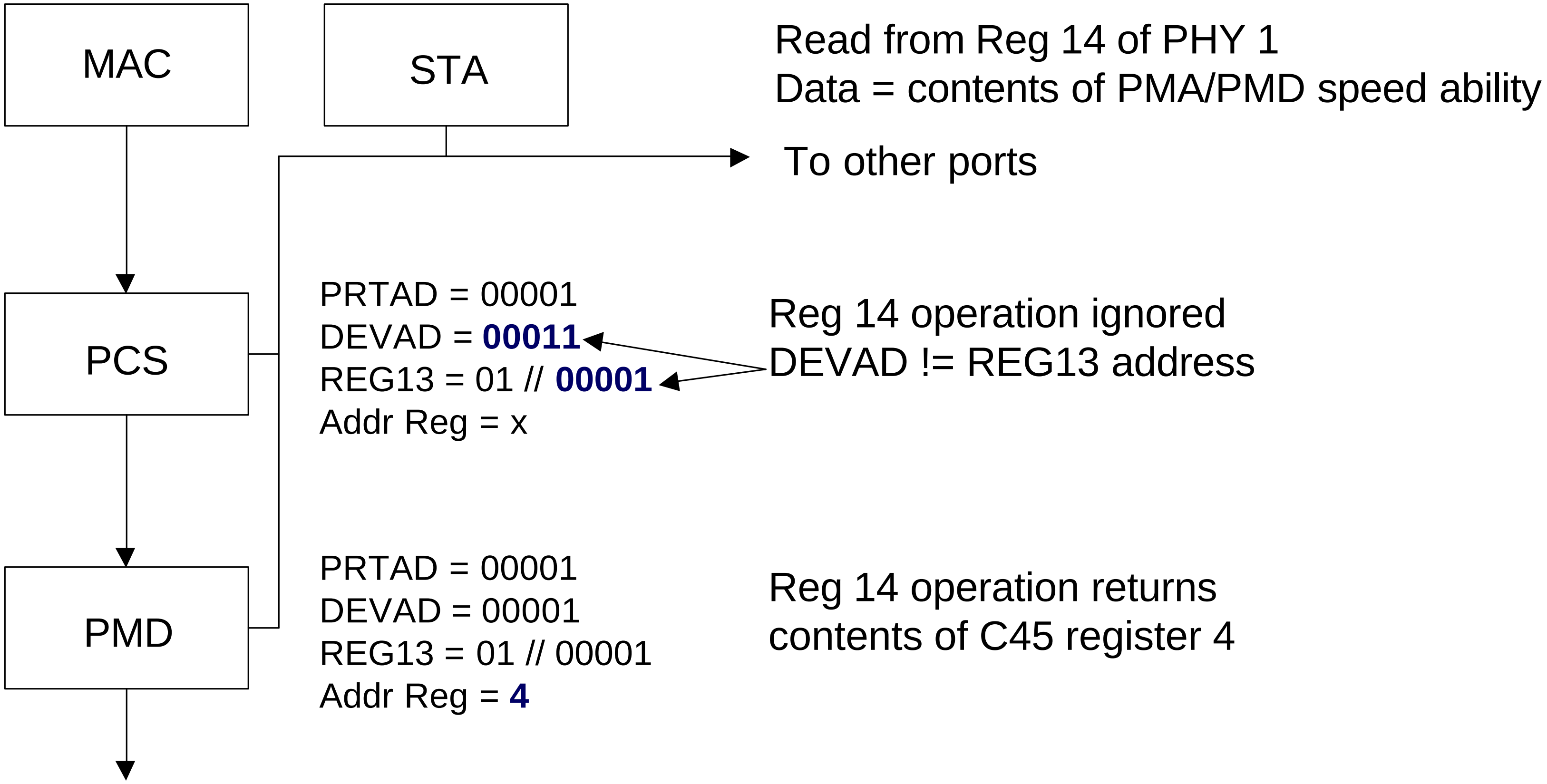
Reg 13 contents stored

PMD

PRTAD = 00001
DEVAD = 00001
REG13 = 01 // 00001
Addr Reg = 4

Reg 13 contents stored

Example - Part 7



MAC

STA

Read from Reg 14 of PHY 1
Data = contents of PMA/PMD speed ability

To other ports

PCS

PRTAD = 00001
DEVAD = **00011**
REG13 = 01 // **00001**
Addr Reg = x

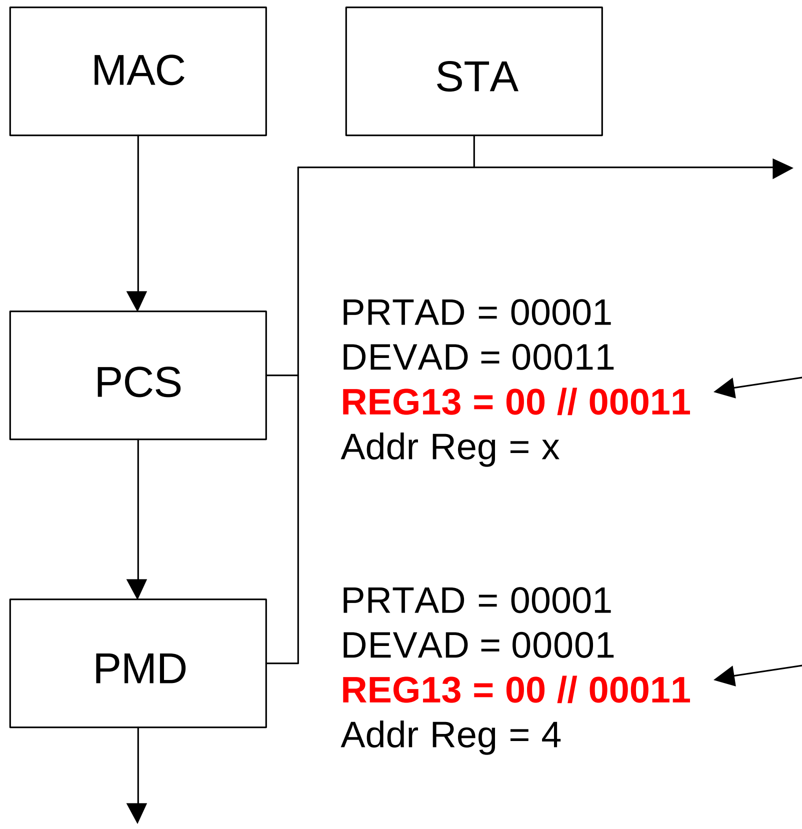
Reg 14 operation ignored
DEVAD != REG13 address

PMD

PRTAD = 00001
DEVAD = 00001
REG13 = 01 // 00001
Addr Reg = **4**

Reg 14 operation returns
contents of C45 register 4

Example - Part 8



Write to Reg 13 of PHY 1
FN = 00 (Address Reg), DEVAD = 3

To other ports

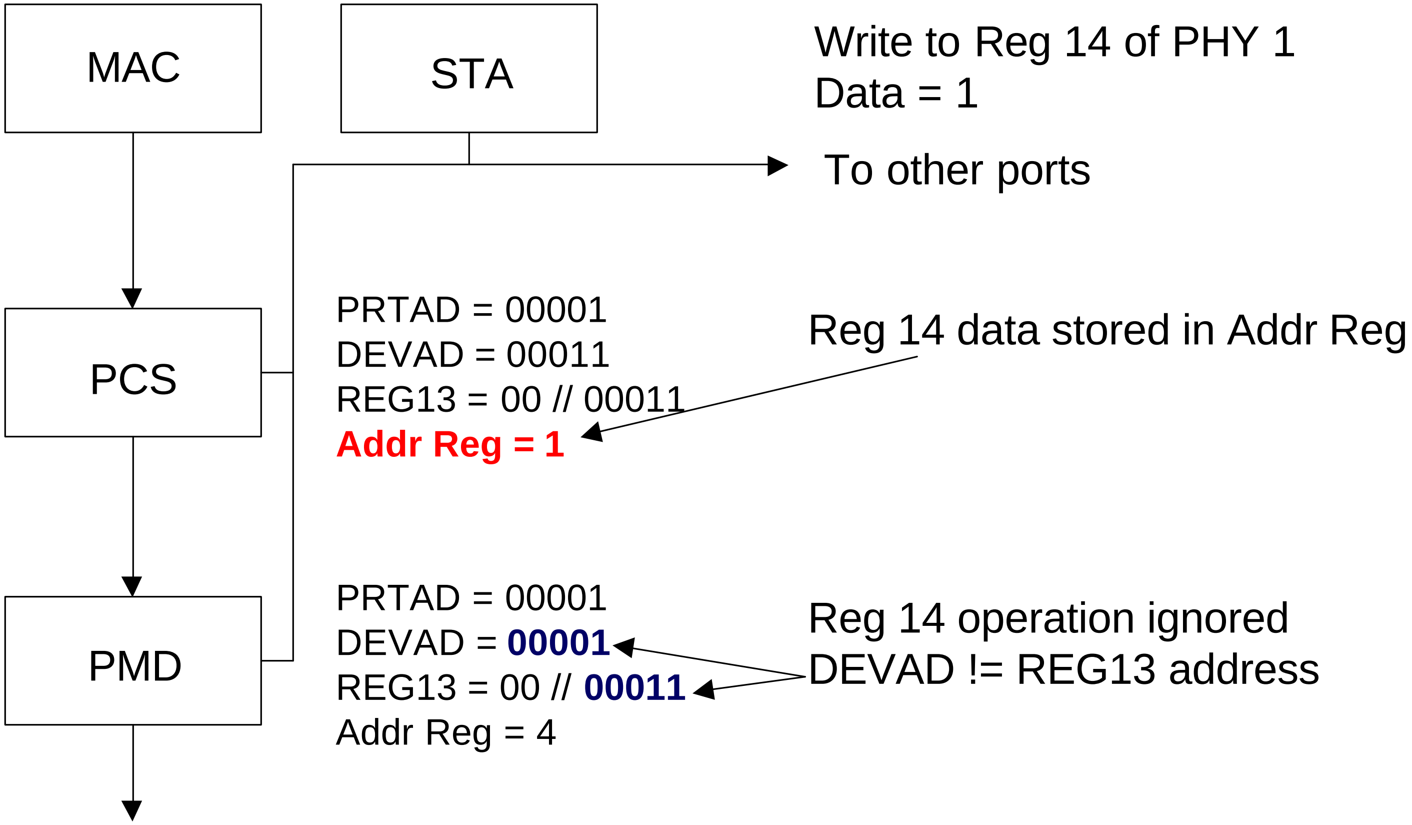
PRTAD = 00001
DEVAD = 00011
REG13 = 00 // 00011
Addr Reg = x

Reg 13 contents stored

PRTAD = 00001
DEVAD = 00001
REG13 = 00 // 00011
Addr Reg = 4

Reg 13 contents stored

Example - Part 9



Write to Reg 14 of PHY 1
Data = 1

To other ports

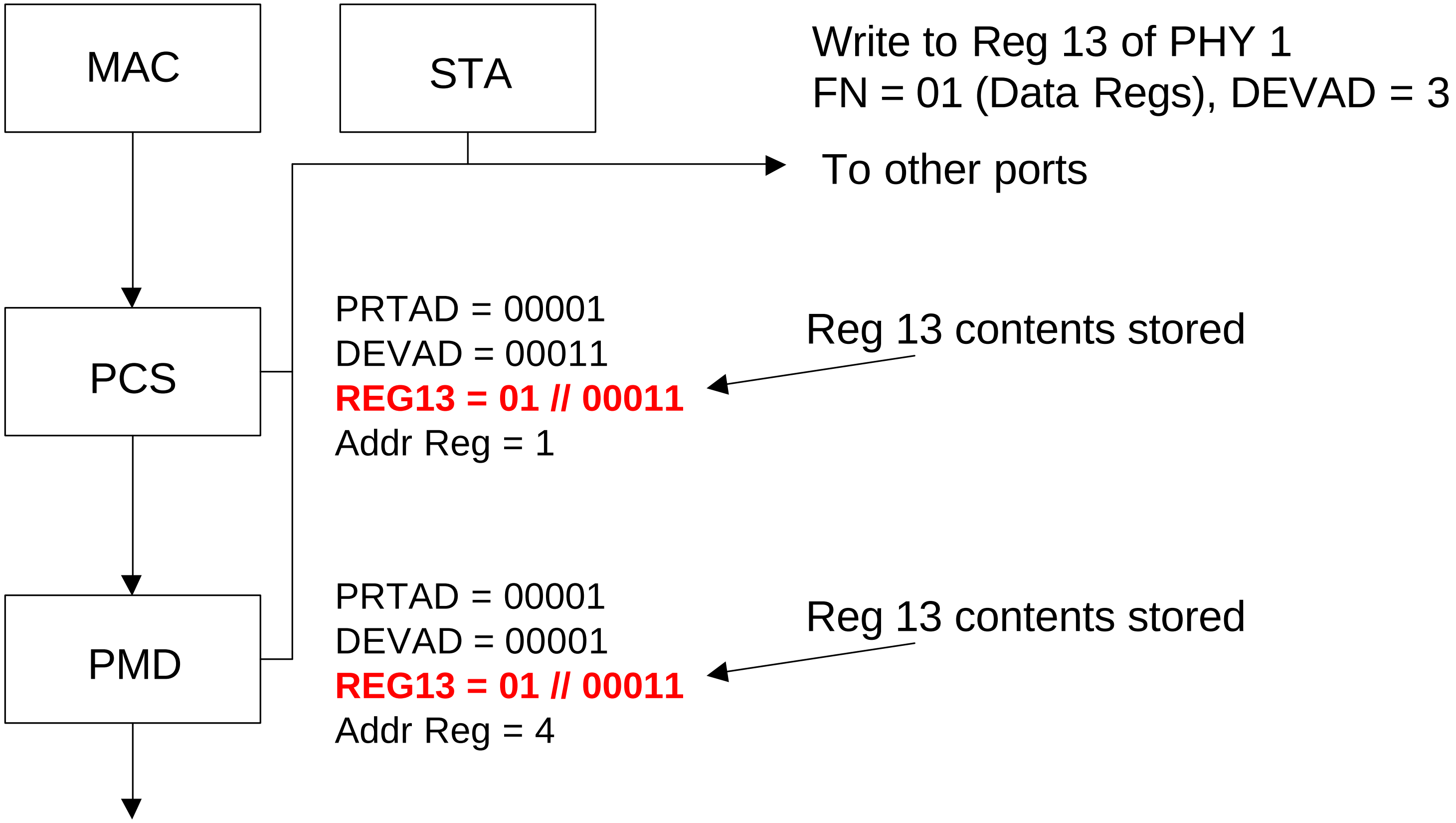
PRTAD = 00001
DEVAD = 00011
REG13 = 00 // 00011
Addr Reg = 1

Reg 14 data stored in Addr Reg

PRTAD = 00001
DEVAD = **00001**
REG13 = 00 // **00011**
Addr Reg = 4

Reg 14 operation ignored
DEVAD != REG13 address

Example - Part 10



MAC

STA

Write to Reg 13 of PHY 1
FN = 01 (Data Regs), DEVAD = 3

To other ports

PCS

PRTAD = 00001
DEVAD = 00011
REG13 = 01 // 00011
Addr Reg = 1

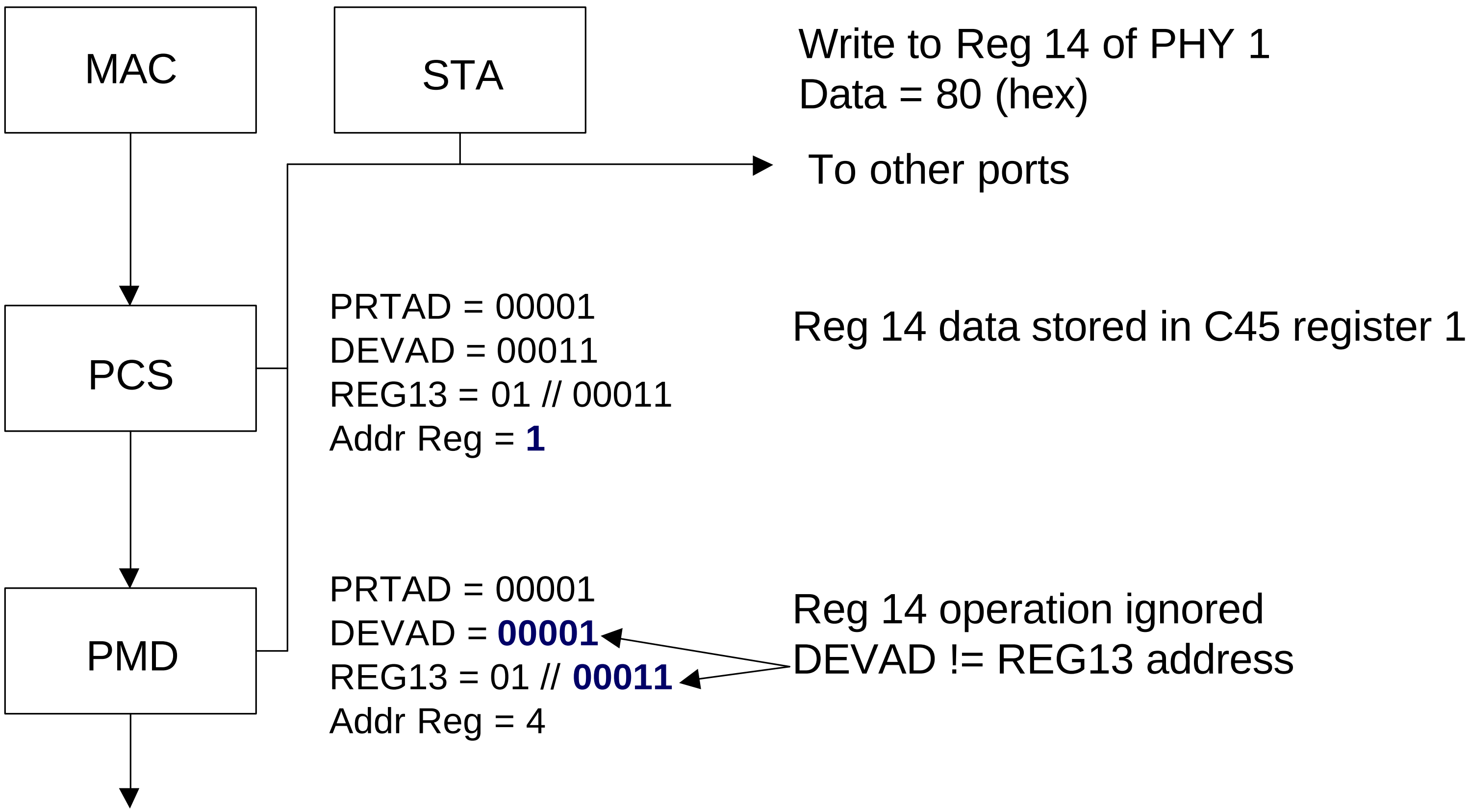
Reg 13 contents stored

PMD

PRTAD = 00001
DEVAD = 00001
REG13 = 01 // 00011
Addr Reg = 4

Reg 13 contents stored

Example - Part 11



Write to Reg 14 of PHY 1
Data = 80 (hex)

To other ports

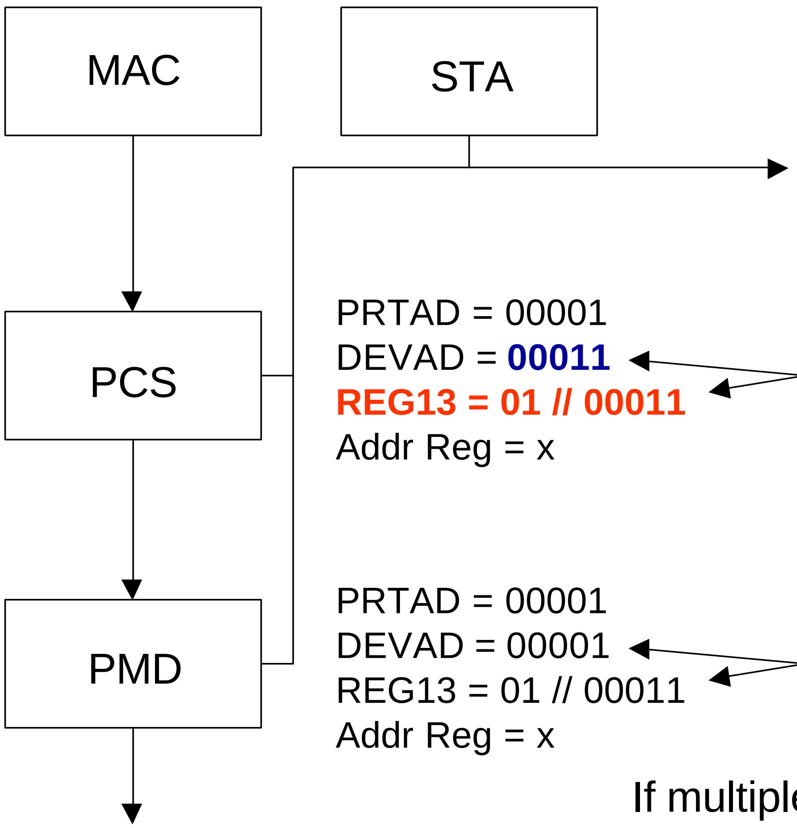
PRTAD = 00001
DEVAD = 00011
REG13 = 01 // 00011
Addr Reg = **1**

Reg 14 data stored in C45 register 1

PRTAD = 00001
DEVAD = **00001**
REG13 = 01 // **00011**
Addr Reg = 4

Reg 14 operation ignored
DEVAD != REG13 address

Example - Part 12



Read from Reg 13 of PHY 1

To other ports

PRTAD = 00001
DEVAD = **00011**
REG13 = 01 // 00011
Addr Reg = x

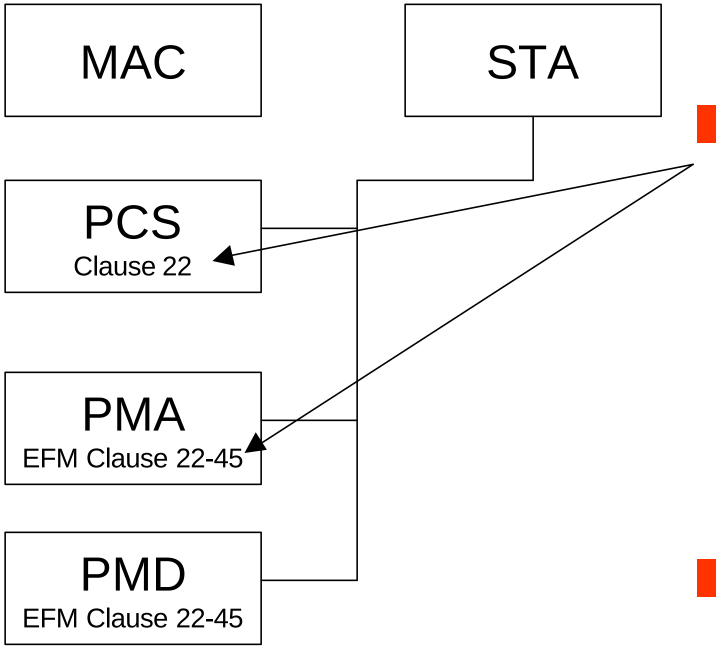
Reg 13 contents ARE returned from the PCS since DEVAD = REG13 address

PRTAD = 00001
DEVAD = 00001
REG13 = 01 // 00011
Addr Reg = x

Reg 13 contents are NOT returned from the PMD since DEVAD != REG13 address

If multiple MMDs responded to a Register 13 Read then multiple MMDs would drive the MDIO line at the same time. So only the PMD whose DEVAD = REG 13 responds to Register 13 Reads

Port Address Issue



- If PCS, PMA and PMD all use the same port address they must all support this proposal
- The Mixed Example Shown will not work if the PCS, PMA and PMD have the same port address
  - The PCS will respond to Reg 13 reads corrupting Reg 13 reads from the PMA and PMD
- Resolution: In implementations where existing Clause 22 MMDs are mixed with new Clause 22 to 45 MMDs, it is required that the MMDs be on different port addresses

Reg 13 Opcode - 1st Try

- Use an Opcode identical to Clause 45’s
  - 00 = Read/Write Address
  - 01 = Read/Write Data
  - 10 = Read/Write Data with post increment
  - 11 = Read/Write Data
- The Options is RED are side effects not supported in Clause 45
- PRO = Uses same Opcodes as Clause 45
- CON = But they don’t work the same due to the side effects & can’t be made to work the same
- CON = Opcodes 10 & 11 are identical

Reg 13 Opcode - 2nd Try

- Use Separate Post Inc & Addr/Data bits
  - 00 = Read/Write Address (no post increment)
  - 01 = Read/Write Data (no post increment)
  - 10 = Read/Write Address (with post increment)
  - 11 = Read/Write Data (with post increment)
- The Opcode in RED is a side effect that doesn’t do anything or is the same as Opcode 00
- PRO = Independent functional bits
- CON = Opcode 10 does not make sense so it is either wasted or reserved

Reg 13 Opcode - 3rd Try

- Call the Opcode bits Function bits as:
  - 00 = Read/Write Address Register
  - 01 = Read/Write Data (no post increment)
  - 10 = Read/Write Data (with post increment on both reads and writes)
  - 11 = Read/Write Data (with post increment on writes only)
- This is what is in the proposal
- PRO = All the functions of Clause 45 plus more
- PRO = FN 11 supports Read/Modify/Writes with post increment
- CON = Different Opcodes from Clause 45
- Solution: They are called FN (function) bits