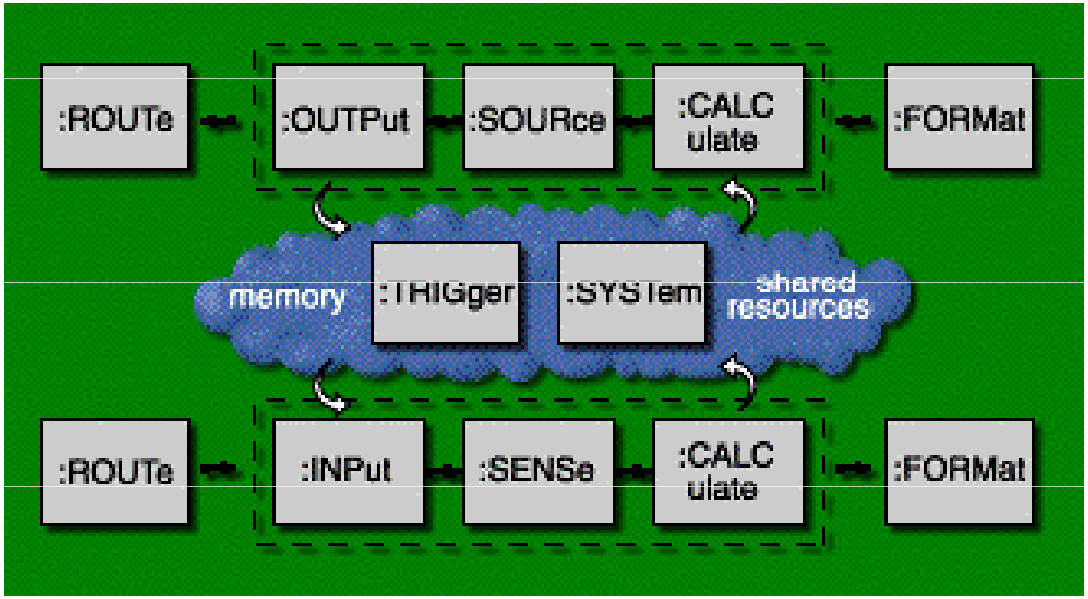
Developing a SCPI command set



Building good "systems citizen" instruments

Agilent Technologies

Agilent Technologies

Relevant Standards

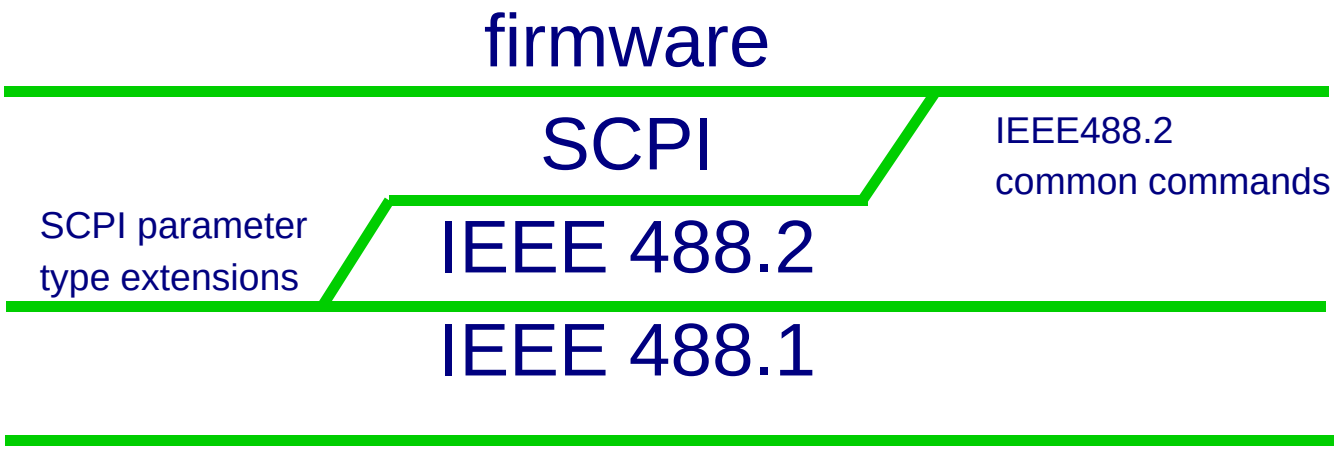
SCPI builds on top of the IEEE 488.1 and 488.2 standards. It is necessary to understand certain aspects of these standards to design a SCPI instrument.

This section describes how these standards interrelate and aspects of 488.1 and 488.2 that are necessary to design a SCPI instrument.

Agilent Technologies

Layering of standards

- SCPI and IEEE 488.2 intimate, crossing boundaries



Agilent Technologies

So what is IEEE 488.1?

- aka: HP-IB, GPIB, "488"
- Transport layer
- Protocol with out of band messages
  - Device clear
  - Service request (Interrupt from device to controller)
  - Serial poll
  - End of message indication ("^END")

Agilent Technologies

So what is IEEE 488.2?

- Standard syntax
  - Commands, Parameters, and Responses
- Builds on 488.1
  - Common status reporting
  - Defines usage of out of band messages
- MEP - Message Exchange Protocol
- Common (* star) commands
- Useful with any ASCII interface
- Required by SCPI

Agilent Technologies

So what is SCPI?

- A Dictionary of instrument commands
  - Based on 488.2 syntax
  - Based on state model of the instrument
  - Hierarchically based
- Some rules
- Terms
  - tree
  - subsystem

Agilent Technologies

SCPI Benefits

- Consistent style
  - No promise of interchangeability
- Remote interface oriented (not front panel)
- Use model consistent across products
- Open ended

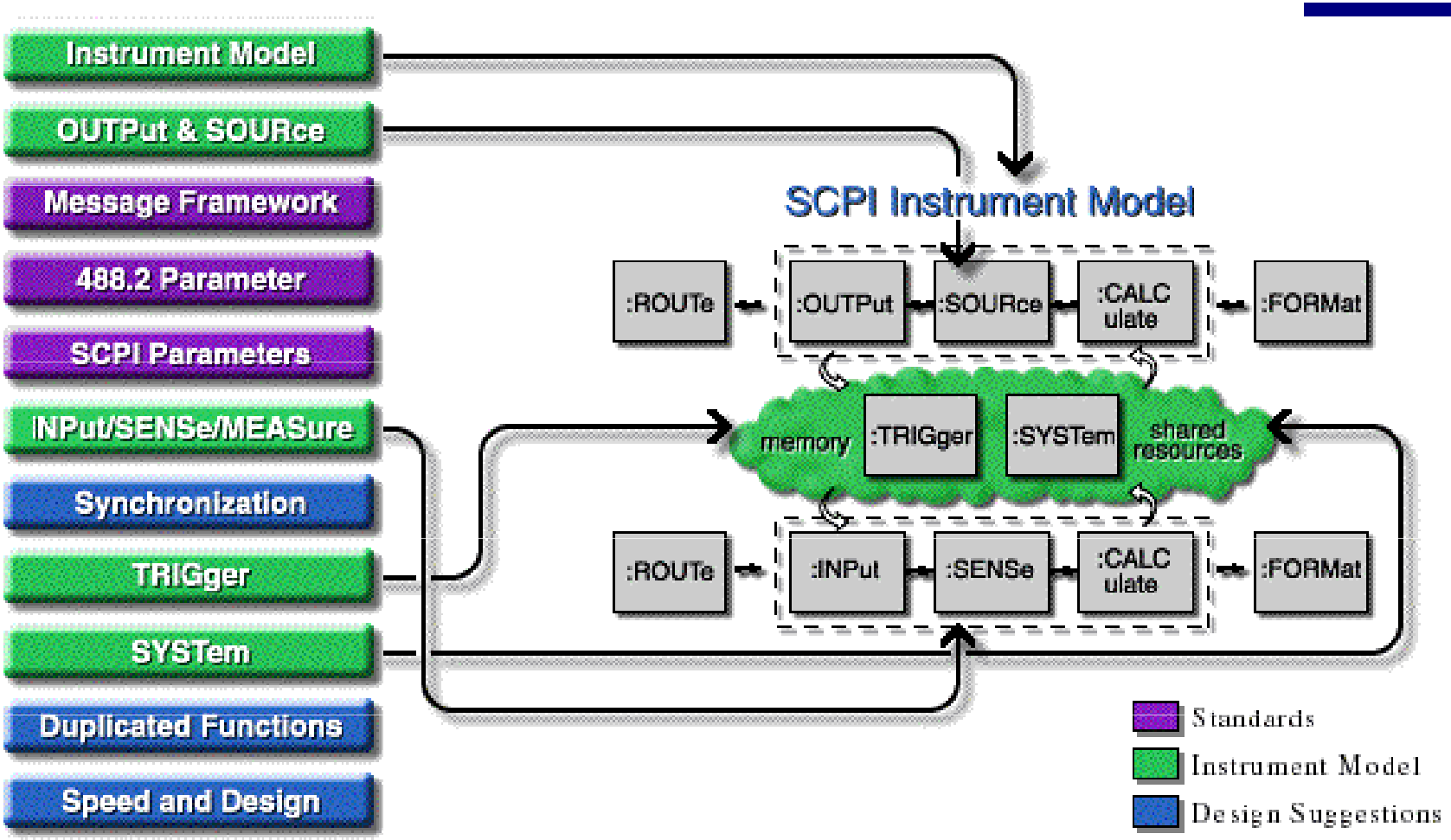
Agilent Technologies

SCPI - Open Ended

- Liberal rules for functionality not covered by standard
- Process for rapid language additions
- Mechanism for defining conformance as standard engulfs undefined functionality

Agilent Technologies

Designing Better Instruments With SCPI



Agilent Technologies

Instrument Model & Mnemonic Generation

Let's talk about the general background for inventing SCPI commands. It consists of the rules and steps for generating mnemonics; the SCPI instrument model, which provides the first step in grouping an instruments functionality, and finally how the command hierarchy works.

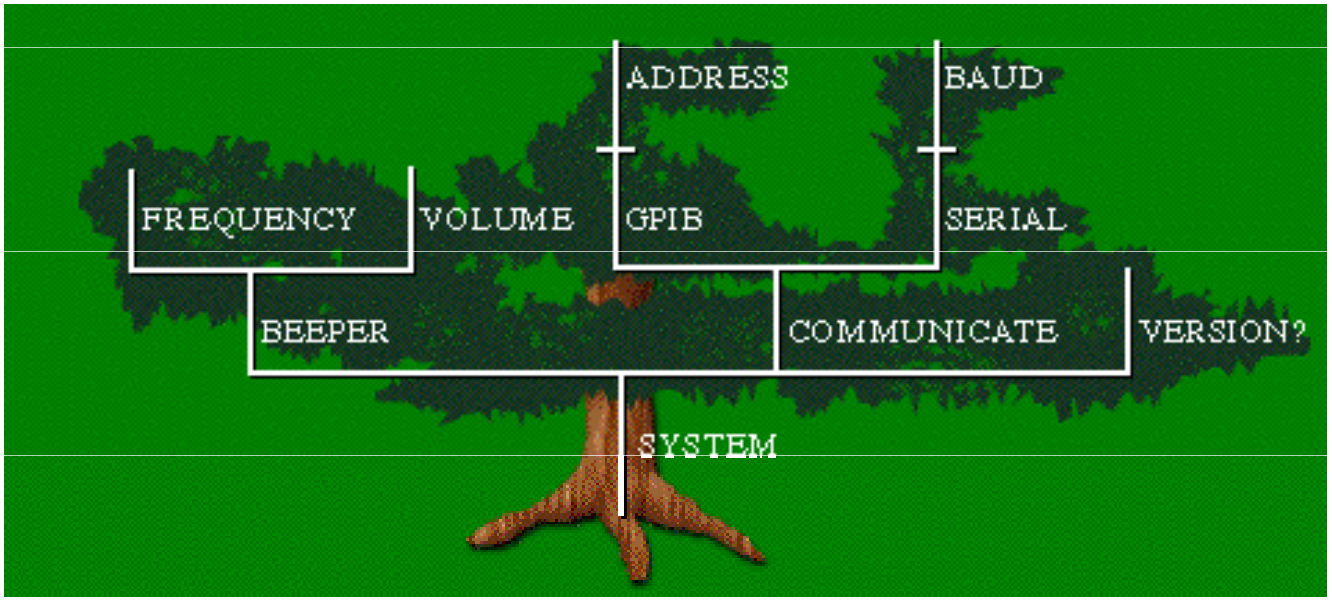
This section also introduces the SCPI Measure concept.

Agilent Technologies

Tree example

Given the following commands

```
SYSTEM:BEEPER:FREQUENCY
SYSTEM:BEEPER:VOLUME
SYSTEM:COMMUNICATE:GPIB:ADDRESS
SYSTEM:COMMUNICATE:SERIAL:BAUD
SYSTEM:VERSION?
```



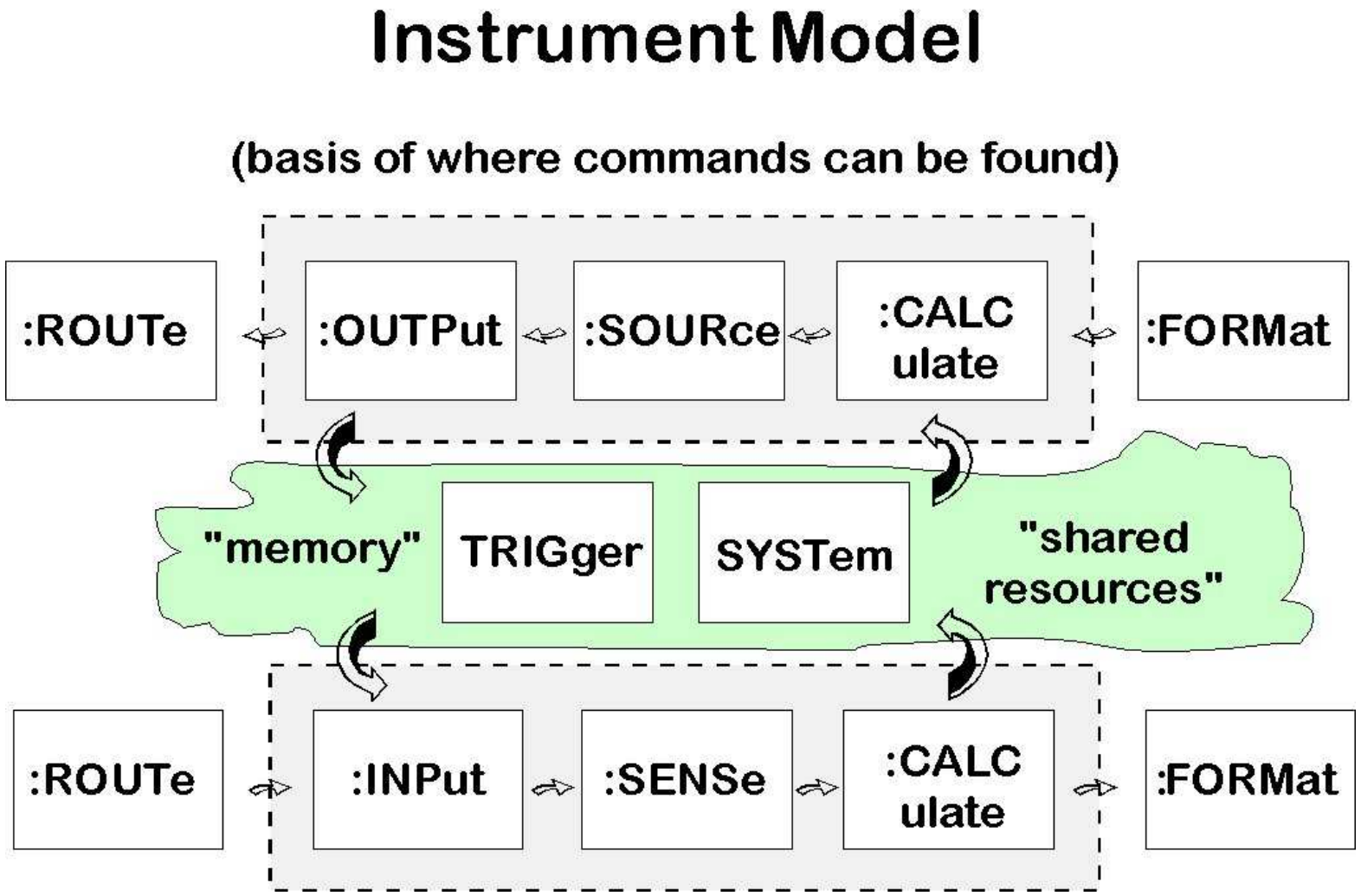
Agilent Technologies

SCPI Instrument Model Features

- Simple model
- Easily sub-setted
- Known data flow
- Predictability - from knowing the concepts
- Model provides:
  - Common view of instrument capabilities
  - Common name for instructions/state variables
    - Mnemonics derived from model

Agilent Technologies

SCPI Instrument Model



Agilent Technologies

SCPI dichotomy - MEASure versus subsystems

- MEASure provides
  - Ease of use - do good
  - Easy default settings
  - Instrument independence
- No source (APPLy) equivalent
- Subsystem commands can be thought of as:
  - names of state variables
  - accessor function entry points
  - queries are commands too

Agilent Technologies

Creating SCPI mnemonics - Rules

- Mnemonics have both long and short forms
- Long form generated from a word or phrase
  - Phrase: long form = 1st letter each word + all of last word
  - Single Word:long form = word
- If long form is 4 characters or less no short form
- To create short form from long form:
  - Short form = first 4 chars
  - Except drop 4th char if it is a vowel
- 12 character limit, including any suffix

Agilent Technologies

Creating SCPI mnemonics - Usual case

	Long form	Short form
STATE	STATE	STAT
COUPLING	COUPLING	COUP
HARD COPY	HCOPY	HCOP
SWEEP	SWEEP	SWE
FREE	FREE	FREE
RELATIVE VELOCITY	RVELOCITY	RVEL
AM	AM	AM
COMMUNICATE	COMMUNICATE	COMM

Agilent Technologies

Creating SCPI mnemonics - Exceptions

- 12 character limit, including any suffix
- Use industry standards where recognized

  E.g., CW, TTLTRG, FM
- What's a vowel? a,e,i,o,u. W? Y?
  - example: "numerical control system"
  - long form: NCSYSTEM
  - short form: NCS or NCSY - do both!

Agilent Technologies

Creating SCPI mnemonics - Style

- 12 character limit, including any suffix
  - try to use 10 or less!
- Choose phrases/words for good mnemonics
  - Usually a well chosen word is best
  - Consider mnemonic a phrase generates
- Use the hierarchy to create context - don't get carried away
- Consider potential need for growth
- Always look for existing patterns
  - E.g., MODE, STATe, AUTO

Agilent Technologies

To locate a command in the SCPI reference manual you need to:

- ☐ Just look it up in the alphabetical listing.
- ☐ Determine where the command falls in the SCPI instrument model, then look in that section of the manual. From that point, look at the subsections within that section for functions that are logically related to the command you are looking for.
- ☐ Flip through the book aimlessly.

Agilent Technologies

To locate a command in the SCPI reference manual you need to:

- ☐ Just look it up in the alphabetical listing.
- ☒ Determine where the command falls in the SCPI instrument model, then look in that section of the manual. From that point, look at the subsections within that section for functions that are logically related to the command you are looking for.
- ☐ Flip through the book aimlessly.

Within each subsection the commands are organized alphabetically, however each of these is again hierarchical.

Therefore to find the right command, you basically need to look through the hierarchy subsequently refining the description of the control you require.

Agilent Technologies

In what section of the SCPI book would you look for a command that controls the symmetry of a generated waveform?

- ☐ INPut
- ☐ OUTPut
- ☐ SENSe
- ☐ SOURce
- ☐ TRIGger

In what section of the SCPI book would you look for a command that controls the hysteresis on the external trigger input?

- ☐ INPut
- ☐ OUTPut
- ☐ SENSe
- ☐ SOURce
- ☐ TRIGger

In what section of the SCPI book would you look for a command to reset the input over-voltage protection circuit?

- ☐ INPut
- ☐ OUTPut
- ☐ SENSe
- ☐ SOURce
- ☐ TRIGger

Agilent Technologies

In what section of the SCPI book would you look for a command that controls the symmetry of a generated waveform?

- [ ] INPut
- [ ] OUTPut
- [ ] SENSe
- [x] SOURce
- [ ] TRIGger

In what section of the SCPI book would you look for a command that controls the hysteresis on the external trigger input?

- [ ] INPut
- [ ] OUTPut
- [ ] SENSe
- [ ] SOURce
- [x] TRIGger

In what section of the SCPI book would you look for a command to reset the input over-voltage protection circuit?

- [x] INPut
- [ ] OUTPut
- [ ] SENSe
- [ ] SOURce
- [ ] TRIGger

Agilent Technologies

What are the short form and long form SCPI mnemonics for:
FREQUENCY

- [ ] FREQUENCY FRE
- [ ] FREQUENCY <no short form>
- [ ] FREQUENCY FREQ
- [ ] FREQUEN FREQ

What are the short form and long form SCPI mnemonics for:
FLOURESCENT

- [ ] FLOURESCENT FL
- [ ] FLOURESCENT FLO
- [ ] FLOURESCENT FLOU
- [ ] FLOURESCENT FLOUR

Agilent Technologies

What are the short form and long form SCPI mnemonics for:
FREQUENCY

- [ ] FREQUENCY FRE
- [ ] FREQUENCY <no short form>
- [x] FREQUENCY FREQ
- [ ] FREQUEN FREQ

Note that SCPI does not allow an instrument to accept partial forms of the header other than the specified short form.

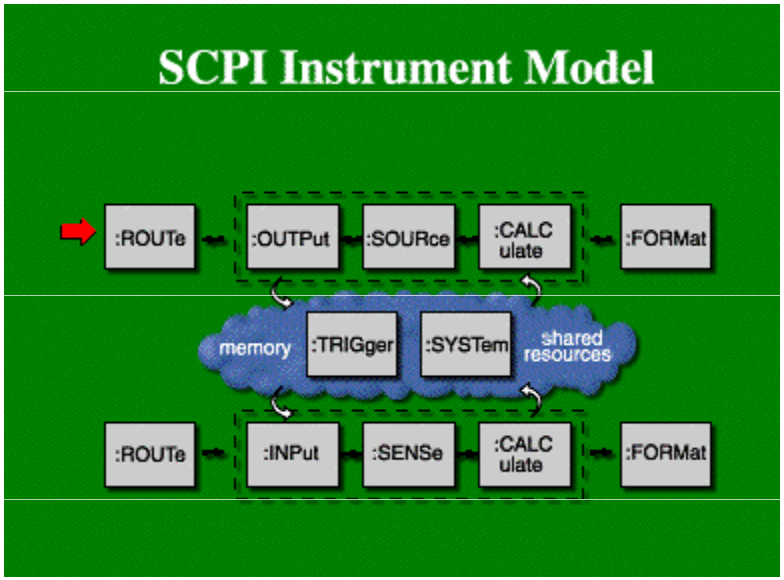
So although the string FREQUEN is unique it is not accepted by a SCPI instrument.

What are the short form and long form SCPI mnemonics for:
FLOURESCENT

- [ ] FLOURESCENT FL
- [x] FLOURESCENT FLO
- [ ] FLOURESCENT FLOU
- [ ] FLOURESCENT FLOUR

Agilent Technologies

OUTPut & SOURce



This section covers the subsections that provide the basic source capabilities. These commands fall in either the source subsystem which controls the actual generation of signals or the output subsystem which provides output control indirectly associated with signal generation.

The output subsystem can also be used as an example of the SCPI hierarchy's use in dealing with functions that may be implemented differently by different classes of instruments.

Agilent Technologies

Instrument Model - OUTPut

- Signal conditioning for "outgoing" signals
  - For example:

    filtering

    attenuation
- All instruments that source a signal have this block
- User programmed values must compensate for the affects of OUTPut

Agilent Technologies

OUTPut Subsystem

```
keyword
     notes
OUTPut
 [:STATe]
     :ATTenuation
     :GAIN
     :PROTection
      [:STATe]
          :DELay
          :TRIPped?
     [query only]
          :CLEar
          [no query]

"OuTpuT:PROT:CLEAR"    "outp?"
```

Agilent Technologies

OUTPut Subsystem - in the beginning

keyword
notes
OUTPut
:FILTer

Agilent Technologies

OUTPut Subsystem - managing growth

```
keyword
     notes
OUTPut
     :FILTer
      [:STATe]
          :FREQuency
          :TYPE
```

Agilent Technologies

OUTPut Subsystem - growing

keyword
notes

```
OUTPut
      :FILTer
       [:LPASs]
      [:STATe]
                  :FREQuency
                  :TYPE
           :HPASs
      [:STATe]
                  :FREQuency
                  :TYPE
```

Agilent Technologies

Instrument Model - SOURce

- Where the signals are generated
- Includes controls for AM, CURRent, DM, FM, FREQuency, FUNCtion, LIST, **PM, PULM**, PULSe, **RESistance**, SWEep, VOLTage
- User programmed values are affected by SOURce multipliers, etc
- SOURce node optional for sources
  - Seems redundant
  - Similar for SENSe and ROUTe

Agilent Technologies

SOURce Subsystem - (1 of 2)

```
keyword
     notes
SOURce
     :POWer
          :ATTenuation
      [:LEVel]
           [:IMMediate]
                [:AMPLitude]
                    :OFFSet
               :TRIGgered
                [:AMPLitude]
                    :OFFSet
```

Agilent Technologies

SOURce Subsystem - (2 of 2)

```
keyword
     notes
SOURce
     :FREQuency
      [:CW|:FIXed]
          :MULTiplier
          :OFFSet
          :STARt
          :STOP
          :CENTer
          :SPAN
```

Agilent Technologies

If you send a source:
SOUR:VOLT 10
OUTP:GAIN 2
What voltage would you measure at the output?

- ☐ 5 volts
- ☐ 10 volts
- ☐ 12 volts
- ☐ 20 volts

If you send a source:
SOURCE:FREQ 1MHZ
SOUR:FREQ:MULTIPLIER 2
SOUR:FREQ:OFFSet 2 MHZ
What frequency would you measure at the output?

- ☐ 1MHZ
- ☐ 2MHZ
- ☐ 4MHZ
- ☐ 6MHZ

Agilent Technologies

If you send a source:
SOUR:VOLT 10
OUTP:GAIN 2
What voltage would you measure at the output?

- [ ] 5 volts
- [ ] 10 volts
- [ ] 12 volts
- [x] 20 volts

If you send a source:
SOURCE:FREQ 1MHZ
SOUR:FREQ:MULTIPLIER 2
SOUR:FREQ:OFFSet 2 MHZ
What frequency would you measure at the output?

- [ ] 1MHZ
- [ ] 2MHZ
- [x] 4MHZ
- [ ] 6MHZ

Notice that the multiplier is applied before the offset.

Agilent Technologies

What is the difference between:
SOUR:POWER:LEVEL:IMMEDIATE:AMPLITUDE 4dBm
SOUR:POWER 4dBm

- ~~The two commands behave identically. The first~~ explicitly includes parts of the hierarchy used for more complex control.
- The first will take effect immediately, while the second will not have any affect till after a trigger.
- ~~The second is syntactically incorrect.~~

What is the difference between:
SOUR:POWER:LEVEL:IMMEDIATE 4dBm
~~SOUR:POWER:LEVEL:TRIG 4dBm~~

- The two commands behave identically. The first explicitly includes parts of the hierarchy used for more complex control.
- ~~The first will take effect immediately, while the~~ second will not have any affect till the instrument receives a trigger.

Which of the following commands set up a source to sweep from 1 MHz to 5 MHZ?

- SOURCE:FREQ:STOP 5MHZ; START 1 MHZ
- SOURCE:FREQ:START 1MHZ; SPAN 4MHZ
- SOURCE:FREQ:CENTER 3MHZ; STOP 5MHZ
- SOURCE:FREQ:CENTER 3MHZ; SPAN 4MHZ
- All of the above.

Agilent Technologies

What is the difference between:
SOUR:POWER:LEVEL:IMMEDIATE:AMPLITUDE 4dBm
SOUR:POWER 4dBm

- [x] The two commands behave identically. The first explicitly includes parts of the hierarchy used for more complex control.
- [ ] The first will take effect immediately, while the second will not have any affect till after a trigger.
- [ ] The second is syntactically incorrect.

What is the difference between:
SOUR:POWER:LEVEL:IMMEDIATE 4dBm
SOUR:POWER:LEVEL:TRIG 4dBm

- [ ] The two commands behave identically. The first explicitly includes parts of the hierarchy used for more complex control.
- [x] The first will take effect immediately, while the second will not have any affect till the instrument receives a trigger.

Which of the following commands set up a source to sweep from 1 MHz to 5 MHZ?

- [ ] SOURCE:FREQ:STOP 5MHZ; START 1 MHZ
- [ ] SOURCE:FREQ:START 1MHZ; SPAN 4MHZ
- [ ] SOURCE:FREQ:CENTER 3MHZ; STOP 5MHZ
- [ ] SOURCE:FREQ:CENTER 3MHZ; SPAN 4MHZ
- [x] All of the above.

Agilent Technologies

IEEE 488.2 Message Framework

IEEE 488.2 defines, at a high level, how messages and responses are constructed and sent between the computer and the instrument. These mechanisms provide a way to send the instrument multiple commands or queries and describes how responses to multiple queries should be separated.

One part of this is the message exchange protocol. The message exchange protocol assures the query result is always fully read by the controller before another command is issued. It also assures that a new command does not interrupt a partially returned response.

Agilent Technologies

IEEE 488.2 Message framework Program Messages

- A <program message> is:
  - `<message unit> { ; <message unit> } <pmt>`
- <PMT> is program message terminator
- Semantics of <message unit> is defined by SCPI, and has the general form:
  - `[:] <header> [?]  [ <parm> { , <parm> } ]`
- <parm> is a parameter (discussed later)
- <header> is a list of mnemonics separated by colons (:)
- '?' denotes a query (required for response)
  - queries may have parameters

Agilent Technologies

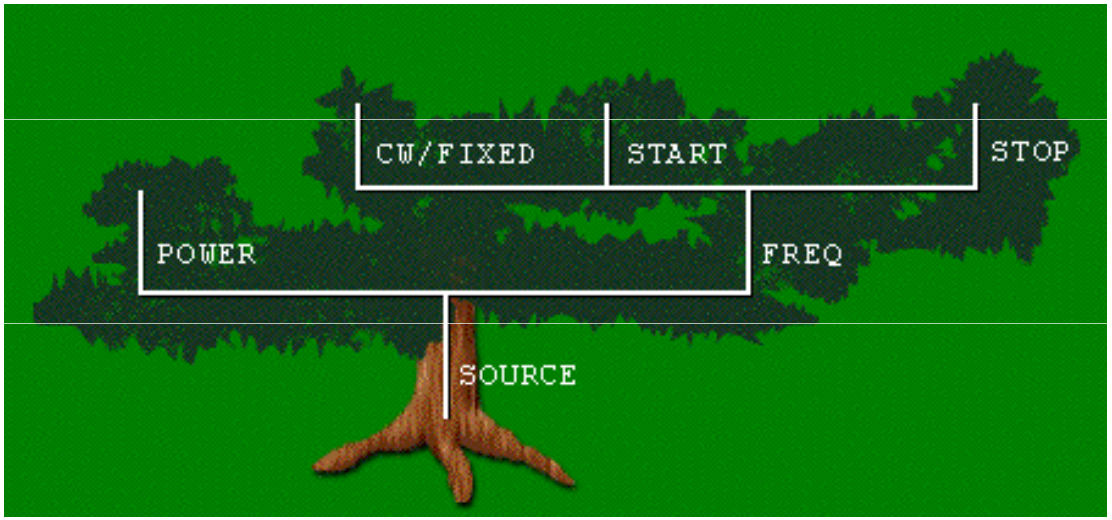
Traversal rules

- Each new message starts parsing from the root of the tree
- Leading colon on message unit resets to root
- Without leading colon, use context of previous command:
  - Simply allows shortcut to programming sibling commands
  - Complicated by default nodes
- star (common) commands don't affect tree position

Agilent Technologies

Compound program message example

```
SOURce
      :FREQuency
            [:CW|:FIXed]
            :STARt
            :STOP
      :POWer
```




```
SOUR:FREQ 5<PMT> SOUR:POW 1<PMT>
:SOUR:FREQ 5;      POW 1 <PMT>
SOUR:FREQ 5;:SOUR:POW 1<PMT>
SOUR:FREQ 5;*WAI;POW 1<PMT>
```

Agilent Technologies

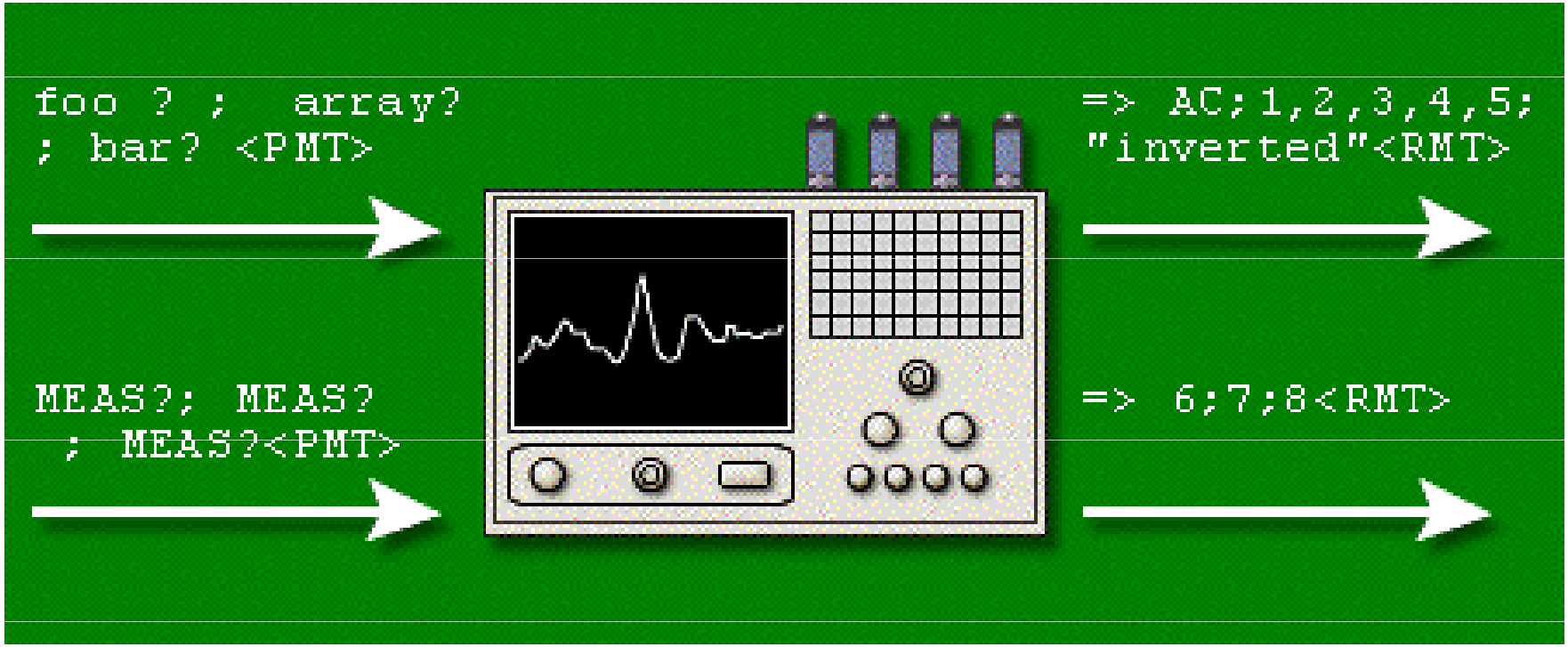
IEEE 488.2 Message framework Response Message

- A program message may have multiple queries. Each query generates a response message unit.
- If a response message unit contains multiple elements then elements are separated by commas (,)
- Response message units are separated by semicolons (;)
- Responses are terminated with <RMT>

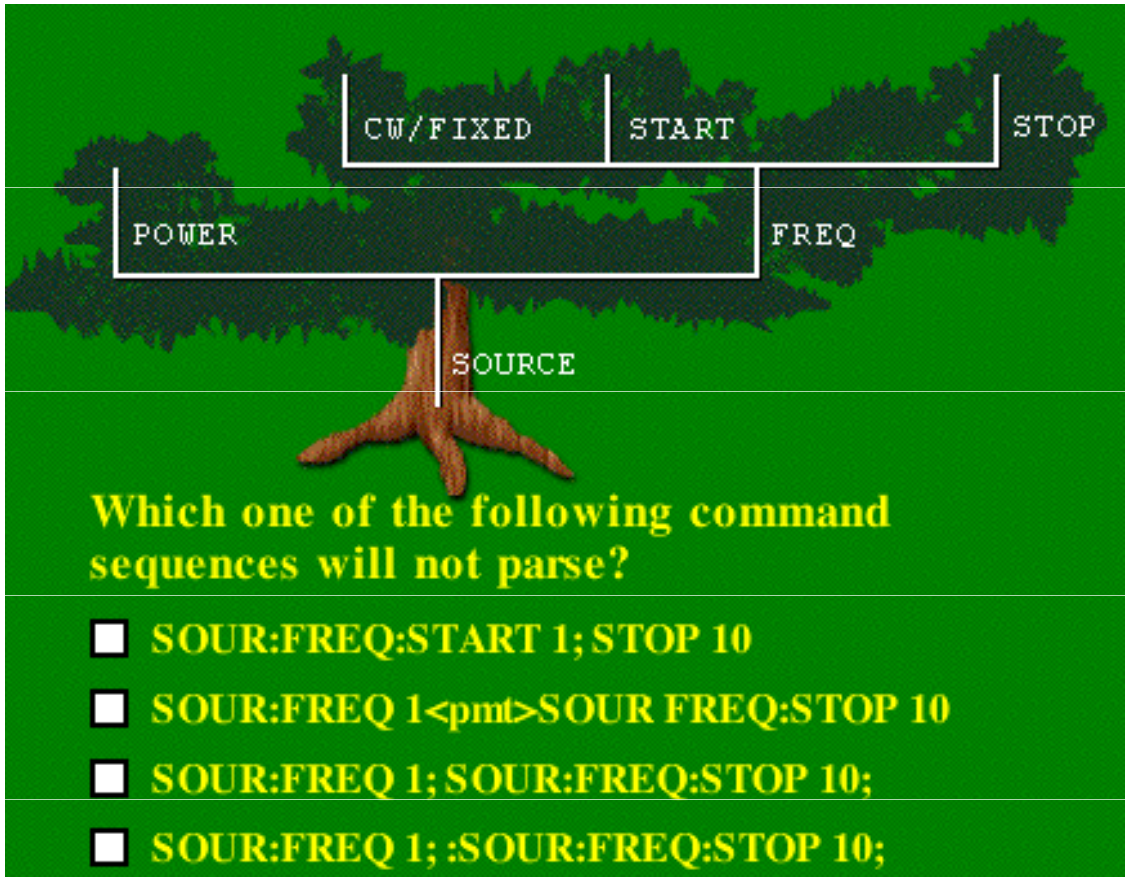
  fancy name for New Line / ^END
- Syntax of response for a given query fixed

Agilent Technologies

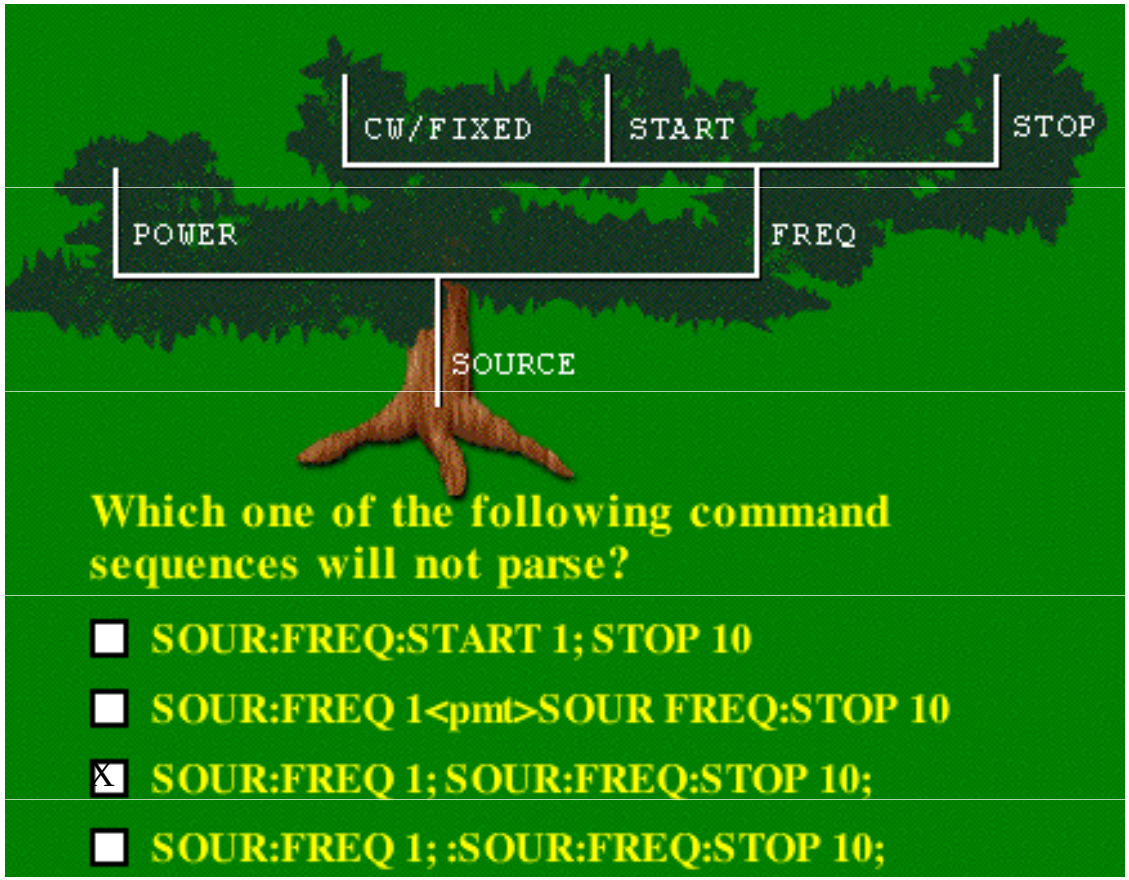
Compound response message example



Agilent Technologies



Agilent Technologies



Agilent Technologies

An unterminated error indicates that:

- ☐ The controller attempted to read from the instrument before sending a complete message.
- ☐ The controller sent a new message to the instrument without reading the previous response.
- ☐ The controller wrote so many queries to the instrument that the output buffer filled up with responses and the input buffer filled up with new queries so that the system has deadlocked.

An interrupted error indicates that:

- ☐ The controller started to read from the instrument before sending a complete message.
- ☐ The controller sent a new message to the instrument without reading the previous response.
- ☐ The controller wrote so many queries to the instrument that the output buffer filled up with responses and the input buffer filled up with new queries so that the system has deadlocked.

A deadlock error indicates that:

- ☐ The controller started to read from the instrument before sending a complete message.
- ☐ The controller sent a new message to the instrument without reading the previous response.
- ☐ The controller wrote so many queries to the instrument that the output buffer filled up with responses and the input buffer filled up with new queries so that the system has deadlocked.

Agilent Technologies

An unterminated error indicates that:

- [X] The controller attempted to read from the instrument before sending a complete message.
- [ ] The controller sent a new message to the instrument without reading the previous response.
- [ ] The controller wrote so many queries to the instrument that the output buffer filled up with responses and the input buffer filled up with new queries so that the system has deadlocked.

An interrupted error indicates that:

- [ ] The controller started to read from the instrument before sending a complete message.
- [X] The controller sent a new message to the instrument without reading the previous response.
- [ ] The controller wrote so many queries to the instrument that the output buffer filled up with responses and the input buffer filled up with new queries so that the system has deadlocked.

A deadlock error indicates that:

- [ ] The controller started to read from the instrument before sending a complete message.
- [ ] The controller sent a new message to the instrument without reading the previous response.
- [X] The controller wrote so many queries to the instrument that the output buffer filled up with responses and the input buffer filled up with new queries so that the system has deadlocked.

Once the instrument has detected the deadlock, the instrument is required to clear the output buffer AND any subsequent responses are cleared until the instrument parses an end of message. In this way, the system never actually deadlocks.

Agilent Technologies

Given the three queries, with corresponding responses:

ABLE? 1

BAKER? 2

CHARLIE? 3,4

What is the response to:

ABLE?;BAKER?;CHARLIE?

- [ ] 1,2,3,4
- [ ] 1;2;3,4
- [ ] 123,4
- [ ] 3,4

Given the three queries, with corresponding responses:

ABLE? 1

BAKER? 2

CHARLIE? 3,4

What is the response to:

CHARLIE?;BAKER?;CHARLIE?

- [ ] 3;4,2,3;4
- [ ] 3,4;2;3,4
- [ ] 3;2;3,4

Agilent Technologies

Given the three queries, with corresponding responses:

ABLE? 1

BAKER? 2

CHARLIE? 3,4

What is the response to:

ABLE?;BAKER?;CHARLIE?

- [ ] 1,2,3,4
- [x] 1;2;3,4
- [ ] 123,4
- [ ] 3,4

Given the three queries, with corresponding responses:

ABLE? 1

BAKER? 2

CHARLIE? 3,4

What is the response to:

CHARLIE?;BAKER?;CHARLIE?

- [ ] 3;4,2,3;4
- [x] 3,4;2;3,4
- [ ] 3;2;3,4

Agilent Technologies

IEEE 488.2 Parameter Types

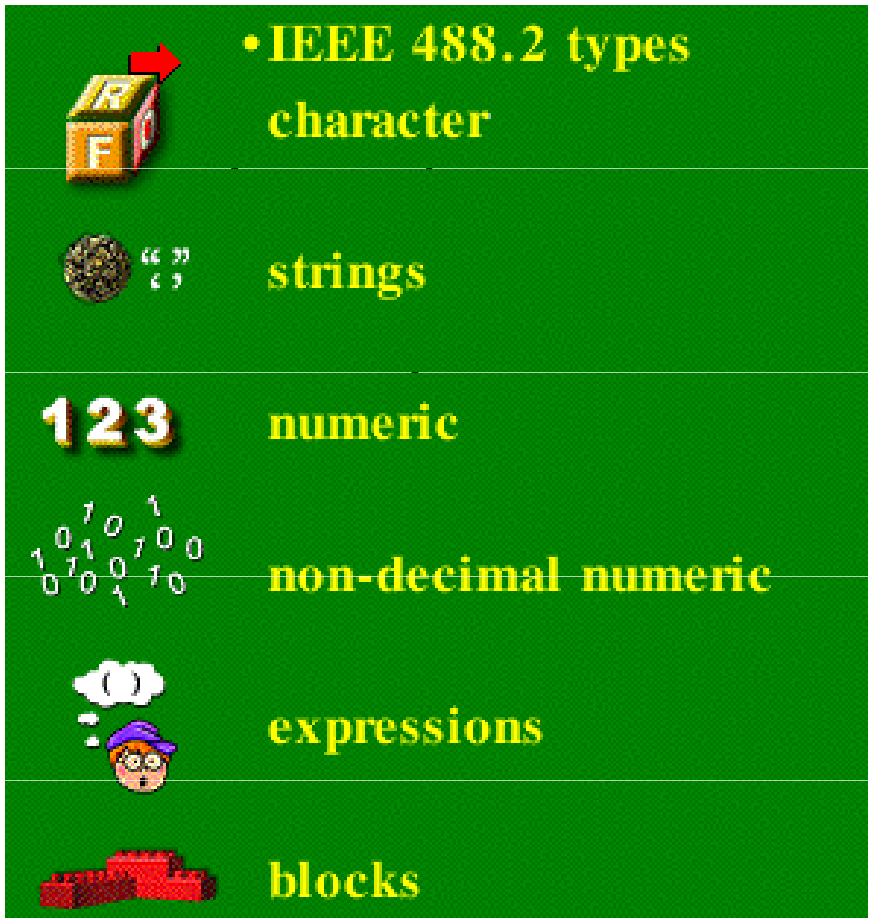
IEEE 488.2 describes the basic syntax of parameters sent to an instrument or returned in a response. This section describes these basic types.

Note that SCPI commands typically build up parameter types that accept more than one 488.2 syntax. For instance the SCPI Boolean parameter accepts either a number (like 0 or 1) or 488.2 character data (like ON or OFF).

Agilent Technologies

Parameters IEEE 488.2

- Forgiving listening, precise talking
- Precise syntax described by IEEE 488.2
- Program and Response formats different



Agilent Technologies

Char

- Character - hello
  - Case Insensitive
  - 12 char limit
  - No embedded spaces
  - response always upper case

send -> hello, HelLo, Hell

receive -> HELL

Agilent Technologies

String

- Parameters:
  - Use single or double quote
  - Embed quotes is string by

    doubling: "Goodbye ""Cruel"" World"

    Or use other type: 'Goodbye "Cruel" World'
- Response always uses double quotes

send -> 'chao "mon" ami'
receive -> "chao ""mon"" ami"

send ->"bye 'my' world"
receive ->"bye 'my' world"

Agilent Technologies

Numeric

- Program data (NRf)
  - Almost any form of decimal number
  - Units also permitted
- Response data (NR1, NR2, NR3)
  - NR1: Signed integer
  - NR2: Float with no exponent (not used by HP)
  - NR3: Float always with exponent
- Units cannot be returned with response
- Form of units defined by IEEE 488.2

Agilent Technologies

Non-decimal numeric

- A byte is transmitted for each digit
- Binary
  - #b010100
- Octal
  - #Q477
- Hexadecimal
  - #h5cFa
- Responses in all upper case
  - eg #H5CFA

Agilent Technologies

Expressions

- Expressions
  - Wrapped in parenthesis '(' and ')'
  - 488.2 allows any syntax which is:
    Valid 488.2 syntax (blocks,strings,etc)
    ASCII that does not resemble 488.2
  - Valid examples:
    ((FB-FA)/2+FA)
    (@1,3:5)
  - Invalid examples:
    (I'll be back)
    (for (i=1;i<10;i++) # iterate over i)

Agilent Technologies

Blocks

- Blocks
  - definite length blocks

    `#14<dab><dab><dab><dab>`

    `#3005<dab><dab><dab><dab><dab>`

    `#14<dab><dab><dab><dab^END>`
  - indefinite length blocks

    `#0<dab><dab><dab><dab>NL^END`

    Improper termination causes command error

    Responses: must be last element in response (use only when necessary)

Agilent Technologies

Arbitrary ASCII

- Response only
- Used by *IDN?
- Only as terminated response message (last query in request)
  - Otherwise generates "Interrupted error"
- Don't use

Agilent Technologies

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ CHAR "foo"
- ☐ NUMERIC 12E17 GHZ
- ☐ BLOCK #111
- ☐ EXPR (foo bar)

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ CHAR foo
- ☐ NUMERIC 12E17 GHZ
- ☐ BLOCK #111
- ☐ NONDECIMAL #O12

Agilent Technologies

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- [x] CHAR "foo"
- [ ] NUMERIC 12E17 GHZ
- [ ] BLOCK #111
- [ ] EXPR (foo bar)

The quotation marks are not allowed in character data.

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- [ ] CHAR foo
- [ ] NUMERIC 12E17 GHZ
- [ ] BLOCK #111
- [x] NONDECIMAL #O12

The character indicating octal is Q.

Agilent Technologies

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ NONDECIMAL #H3ffF
- ☐ NUMERIC 1e1
- ☐ STRING "Never eat soggy waffle's"
- ☐ STRING "Never eat soggy waffle" "s"

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ BLOCK #0abcdef<END> (end asserted with 'f' byte)
- ☐ BLOCK #500000xyzzy<END> (end asserted with 'y')
- ☐ NOND #Q123770
- ☐ NOND #B10010110001010010001001000011

Agilent Technologies

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ NONDECIMAL #H3ffF
- ☐ NUMERIC 1e1
- ☐ STRING "Never eat soggy waffle's"
- ☒ STRING "Never eat soggy waffle" "s"

Spaces are not allowed between the double double quotes.

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ BLOCK #0abcdef<END> (end asserted with 'f' byte)
- ☒ BLOCK #500000xyzzy<END> (end asserted with 'y')
- ☐ NOND #Q123770
- ☐ NOND #B10010110001010010001001000011

Agilent Technologies

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ BLOCK #16abcdef<END> (end asserted with 'f' byte)
- ☐ BLOCK #3abc
- ☐ NOND #HFeedAbe
- ☐ NOND #B10010110001010010001001100011

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ STRING """""""
- ☐ STRING "gaultheria procumbens"
- ☐ EXPR (gaultheria procumbens)
- ☐ EXPR (514 W. 4th St; Apt #17)

Agilent Technologies

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ BLOCK #16abcdef<END> (end asserted with 'f' byte)
- ☒ BLOCK #3abc
- ☐ NOND #HFeedAbe
- ☐ NOND #B10010110001010010001001100011

A correct format for this block would be:
BLOCK #203abc

Consider the instructions:

CHAR	accepts a character parameter
STRING	accepts a string parameter
NUMERIC	accepts a numeric parameter
EXPR	accepts an expression parameter
BLOCK	accepts a block
NONDECIMAL	accepts non-decimal data

Which is illegal?

- ☐ STRING """""""
- ☐ STRING "gaultheria procumbens"
- ☐ EXPR (gaultheria procumbens)
- ☒ EXPR (514 W. 4th St; Apt #17)

Agilent Technologies

SCPI Parameters

The SCPI parameters take the basic syntax defined by IEEE 488.2 for parameters and create more complex types that are easier to use and provide some additional capabilities.

For instance, the SCPI Boolean parameter accepts either a number (like 0 or 1) or 488.2 character data (like ON or OFF).

This section describes the SCPI parameter types and how units are used with numeric values in SCPI.

Agilent Technologies

SCPI Parameter type Extensions to 488.2

- Increase level of forgiving listening
- Talking remains precise
- SCPI types
  - enums
  - <Boolean>
  - <numeric_value>
- Other 488.2 types used as needed
  - Expressions - channel lists, SCPI - DIF
  - Blocks - binary data transfer

Agilent Technologies

SCPI enums and <Boolean>

- Enums
  - use character data and follow long/short form rules
  - responses return short form
- <Boolean>
  - accepts char data of ON or OFF, in any case
  - accepts NRf rounds to integer, then non-zero is on, and zero off
  - responses return 0 or 1

Agilent Technologies

SCPI <numeric_value>

- Required to accept
  - NRf numbers
  - Character data

    INFinity, NINFinity, NAN, in addition to equivalent values (9.9E37, -9.9E37, and 9.91E37)

    MINimum & MAXimum

    DEFault (optional)

    UP & DOWN (optional)

Agilent Technologies

MINimum and MAXimum

- REQUIRED where meaningful
- Set "foo" to lowest value possible currently:
  - `FOO MiniMum <PMT> -or- FOO MIN <PMT>`
- Set "foo" to highest value currently possible:
  - `FOO MAXIMUM <PMT> FOO max <PMT>`
- Ask for highest, but do NOT change value:
  - `FOO? MAX`
- MIN and MAX may depend upon instrument state

Agilent Technologies

DEFault and stepping

- DEFault
  - May optionally be implemented, no query
  - Sets state variable to a reasonable value
  - Often fixed
  - May not query (e.g., FOO? DEF is invalid)
- UP and DOWN
  - May optionally be implemented, no query
  - Floating subsystem

```
...:STEP
     [:INCRement]  <numeric_value
      :PDECade     <numeric_value
      :MODE        LINear | LOG | L125 | L3
```

Agilent Technologies

SCPI <numeric_value> Good practices

- Accept non-decimal numerics
- Allow tolerance bands on limits
- Allow rounding where it makes sense
  - Often at one end of a limit
  - In conjunction with enumerated numerics
- When formatting responses, control precision
  - What does your customer expect
  - Definite length block analogy

Agilent Technologies

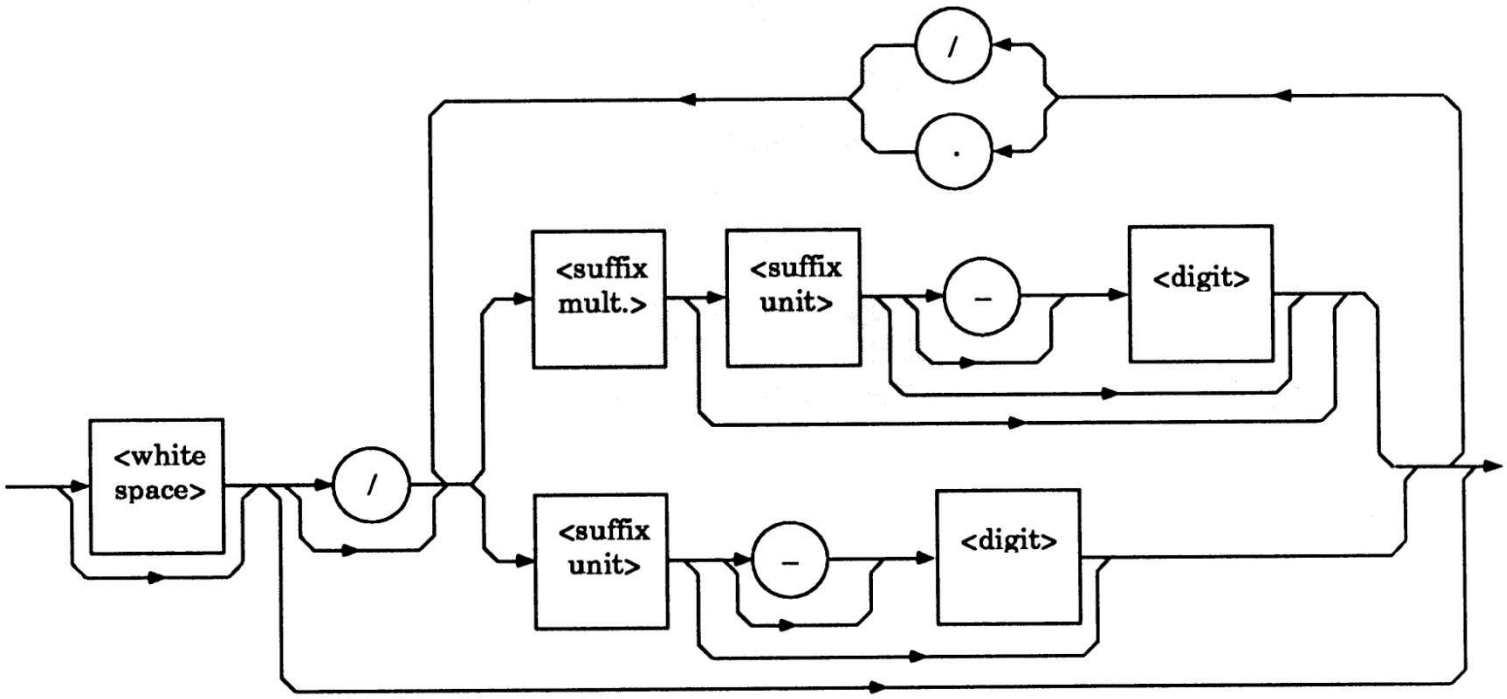
Units in SCPI

- Accepting Units is optional
- If accepted, then case insensitive
- UNIT never returned in query of numeric value
- Default units are used:
  - On input, when units not specified
  - On response (although not sent (SCPI rule))
- Floating UNIT subsystem exists
  - At the root it controls units for whole instrument
    - OUTP:GAIN & SOUR:POW example
  - Any other part of the tree
    - Affects commands below in hierarchy

Agilent Technologies

Units syntax

- Responses don't allow the white space



Agilent Technologies

Floating UNIT subsystem

```
:UNIT
  :ANGLe          DEG|RAD
  :CURRent<amplitude_unit>
  :POWer          <amplitude_unit>
  :TEMPerature C|CEL|F|FAR|k
  ...

:UNIT
  :POWER
:DISPlay
 [:WINDow]
    :X
     [:SCALe]
        :UNIT
          :POWer
```

Agilent Technologies

Consider the instructions:

ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is illegal?

- ☐ BOOLEAN ON
- ☐ BOOLEAN 0
- ☐ BOOLEAN 42
- ☐ BOOLEAN #H1

Consider the instructions:

ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is illegal?

- ☐ ENUM BAKer
- ☐ ENUM baker
- ☐ enum BaKeR
- ☐ ENUM "BAKER"

Agilent Technologies

Consider the instructions:

ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is illegal?

- [ ] BOOLEAN ON
- [ ] BOOLEAN 0
- [ ] BOOLEAN 42
- [x] BOOLEAN #H1

Consider the instructions:

ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is illegal?

- [ ] ENUM BAKer
- [ ] ENUM baker
- [ ] enum BaKeR
- [x] ENUM "BAKER"

Agilent Technologies

Consider the instructions:
ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is illegal?

- [ ] ENUM BAKer
- [ ] ENUM baker
- [ ] enum BaKeR
- [ ] ENUM "BAKER"

Consider the instructions:
ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is different from the others?

- [ ] NUMERIC 1 kv/us
- [ ] NUMERIC 1 kv,mhz
- [ ] NUMERIC 1E-6 kv/s
- [ ] NUMERIC 1000 v/us
- [ ] numeric 1E9 v/s

Agilent Technologies

Consider the instructions:
ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is illegal?

- [ ] ENUM BAKer
- [ ] ENUM baker
- [ ] enum BaKeR
- [X] ENUM "BAKER"

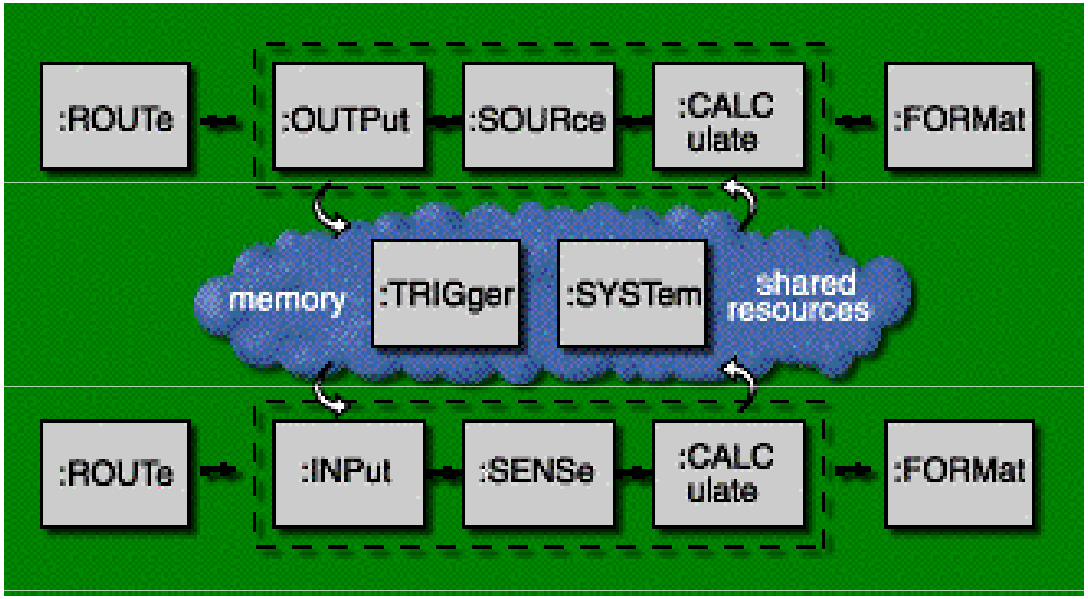
Consider the instructions:
ENUM accepts: {ABLE, BAKer, CHARlie}
NUMBER accepts normal SCPI numbers with units
BOOLEAN accepts a SCPI Boolean

Which is different from the others?

- [ ] NUMERIC 1 kv/us
- [ ] NUMERIC 1 kv,mhz
- [X] NUMERIC 1E-6 kv/s
- [ ] NUMERIC 1000 v/us
- [ ] numeric 1E9 v/s

Agilent Technologies

INPut / SENSe / MEASure / Format



The section describes the SCPI subsystems that are directly involved in making measurements and formatting responses.

It includes the input and sense subsystems that controller sensors, and the measure commands that provide a portable, easy to use alternative for measuring instruments.

Agilent Technologies

Instrument model - INPut

- Signal conditioning for "incoming" signals
  - For example:
    filtering
    biasing
    attenuation
- All sense instruments have this block
- User programmed values must compensate for the effects of INPut

Agilent Technologies

INPut Subsystem

```
keyword
     parameters
INPut
 [:STATe] <Boolean>
     :ATTenuation <numeric_value>
     :COUPling
     AC|DC|GROund
     :FILTer
      [:STATe] <Boolean>
          :AUTO <Boolean>|ONCE
```

Agilent Technologies

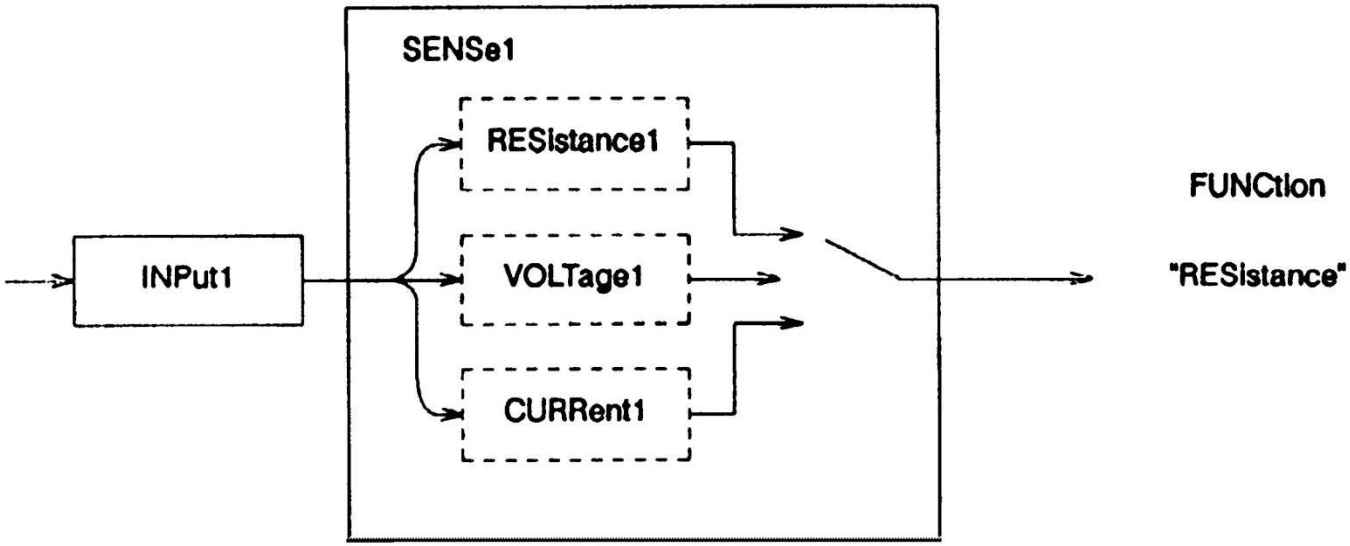
Instrument model - SENSe

- Where the signals are measured
- Command structure analogous to SOURce
- Place to commonly extract data from
  - Remote interface or
  - subsequent processing (e.g., CALCulate)
- Includes controls for AM, CURRent, DM, FM, FREQuency, FUNCtion, LIST, **PM, PULM**, PULSe, RESistance, SWEep, VOLTage
- User programmed values are affected by SENSe multipliers, etc
- SENSe node optional for sensors

Agilent Technologies

Getting measured data out - SENSe:DATA?

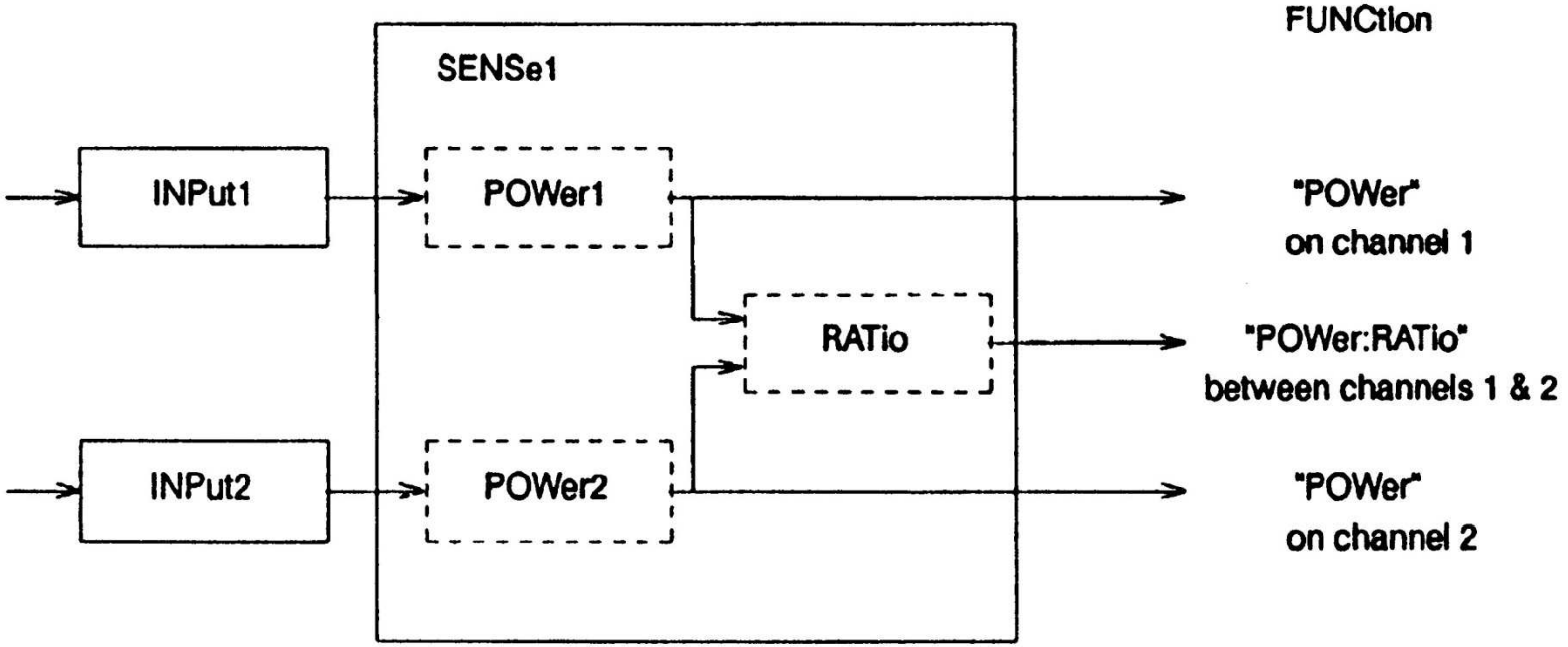
- SENSe:FUNCtion and SENSe:DATA? close friends
  - Example




```
SENS:FUNC "resistance"
INITIATE
SENS:DATA?
```

Agilent Technologies

Getting measured data out
- with concurrent measurements



FUNC:CONC on; ON "pow 1"; ON "pow:ratio 1,2"

INIT;: FUNC:COUNT?; ON?; :DATA?

Agilent Technologies

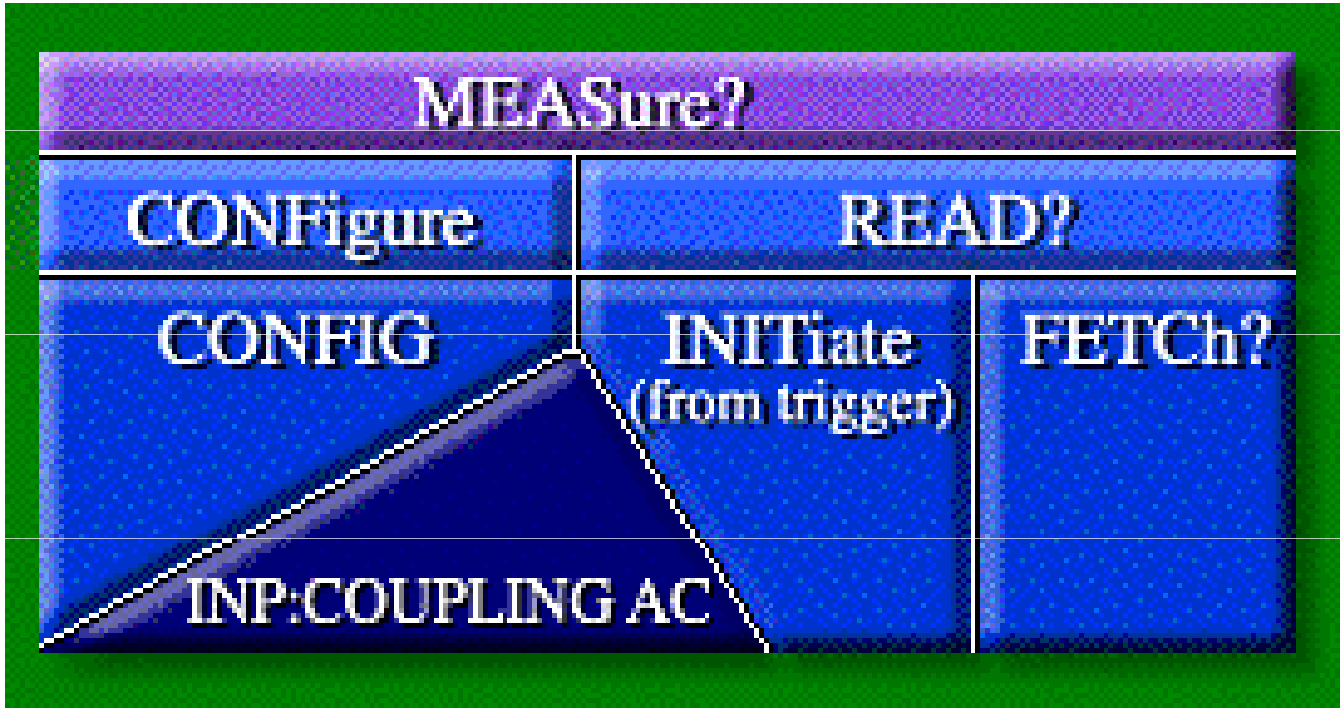
Getting measured data out - MEASure?

- The "do good" action and query
- Simple to use
- Touches all parts of instrument state
  - A given MEAS? always yields same result independent of instruments earlier state
  - Cannot be used for incremental programming
- Currently no syntax for concurrent sensing
  - Can get different aspects of acquired data

Agilent Technologies

MEASure? layering

- A device may provide more granularity of control:



Agilent Technologies

MEASure?

- Note CONFig also has a query
- Error handling - try, but also warn
  - too many parameters
  - can't respect parameter

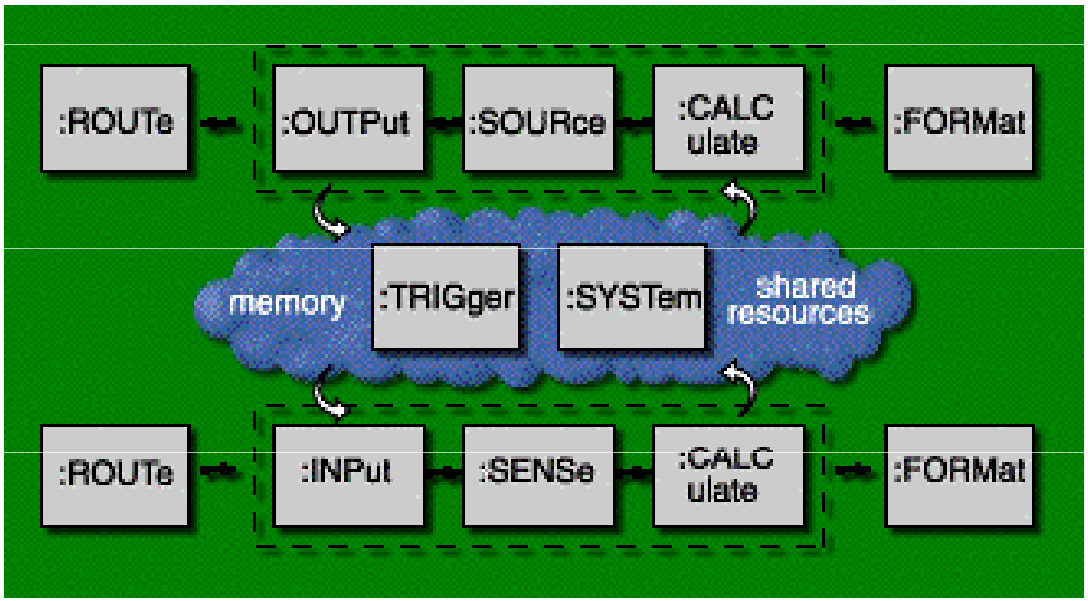
Agilent Technologies

MEASure? and default parameters

```
MEASure
 :VOLTage?[<expected_val>[,<res>]]

MEAS:VOLT? 15V, 0.001V
MEAS:VOLT? 15V
MEAS:VOLT? DEFAULT, 0.001V
```

Agilent Technologies

FORMat Subsystem



- Provides opportunity to match byte ordering to host
  - normal network order (big endian)
  - swapped PC format
- Allow binary (and other) formats of data to and from host

Agilent Technologies

Where would you look for the commands that control the input protection circuit?

- ☐ OUTPut
- ☐ SOURce
- ☐ INPut
- ☐ SENSe:FUNCtion
- ☐ SENSe:FREQuency

Where would you look for the commands that control the current source for a four-wire ohms resistance measurement?

- ☐ OUTPut
- ☐ SOURce
- ☐ INPut
- ☐ SENSe:CURRent
- ☐ SENSe:FRESistance

How do the following differ?

```
MEASURE:VOLTAGE? 10;
CONFIGURE:VOLTAGE 10; READ?
CONFIGURE:VOLTAGE 10; INITIATE ; FETCH?
```

- ☐ The third version allows the user to also read back the AC voltage with a subsequent FETCH.
- ☐ The second version does not trigger the measurement
- ☐ They are all the same.

Agilent Technologies

Where would you look for the commands that control the input protection circuit?

- ☐ OUTPut
- ☐ SOURce
- ☐ INPut
- ☐ SENSe:FUNCtion
- ☒ SENSe:FREQuency

Where would you look for the commands that control the current source for a four-wire ohms resistance measurement?

- ☐ OUTPut
- ☐ SOURce
- ☐ INPut
- ☐ SENSe:CURRent
- ☒ SENSe:FRESistance

Since this source is intimately associated with the sensing function, the control is associated with the sense function.

How do the following differ?

MEASURE:VOLTAGE? 10;
CONFIGURE:VOLTAGE 10; READ?
CONFIGURE:VOLTAGE 10; INITIATE ; FETCH?

- ☐ The third version allows the user to also read back the AC voltage with a subsequent FETCH.
- ☐ The second version does not trigger the measurement
- ☒ They are all the same.

Synchronization & Required 488.2 Commands

IEEE 488.2 defines several commands. Most can be easily understood from the 488.2 specification.

This section lists the 488.2 common commands that must be implemented by SCPI instruments and describes the tools in IEEE 488.2 and SCPI for synchronization between the isntrument and the controller.

Agilent Technologies

SCPI mandated common commands

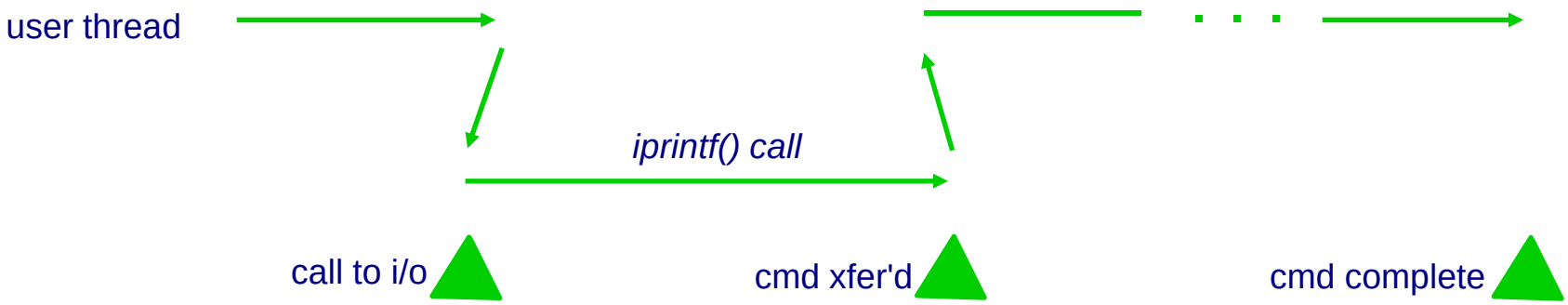
```
*CLS          clear status command
*ESE          std. event status enable command
*ESE?         std. event status enable query
*ESR?         std. event status register query
*IDN?         Identification query
*OPC          operation pending complete command
*OPC?         operation pending complete query
*RST          reset command
*SRE          service request enable command
*SRE?         service request enable query
*STB?         read status byte query
*TST?         self test query
*WAI          wait-to-continue command

OF INTEREST
*TRG          bus trigger command
*SAV          save stored settings command
*RCL          recall stored settings command
```

Agilent Technologies

Synchronization
- Normal behavior

- + Customer computer not blocked
- - When to get data

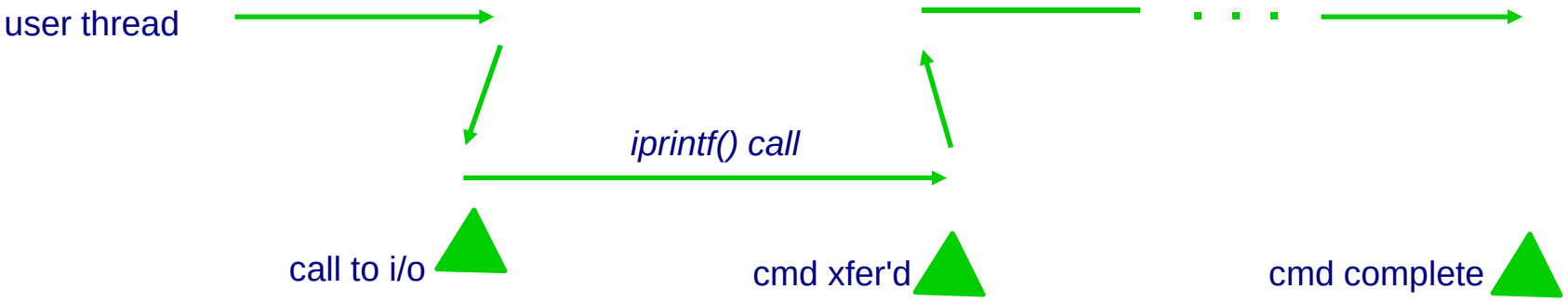
iprintf("MEAS? power:achannel");



Agilent Technologies

Synchronization
- Compound message behavior

iprintf("FOO; :BAR");



No overlapping;
commands serialized

[---------- FOO ----------] [--------- BAR ----------]

Commands parsed serially,
but execution routines handled
in separate threads

[---------- FOO ----------]

[---------- BAR ----------]

Agilent Technologies

Notification of completion - *OPC and *OPC?

- *OPC?
  - MMEM:PACK; *OPC? <PMT>
    computer "reads" the instrument (bus hangs),
    device returns 1 when operation complete,
    then computer sends
    MEAS? volt <PMT>
- *OPC
  - MMEM:PACK; *OPC
    computer waits/polls for "operation complete"
    in Standard Event Status, then sends
    MEAS? volt <PMT>
- Must accept these commands,
- Extra work if you implement overlapping
- Don't use in middle of program message

Agilent Technologies

Forcing synchronization with *WAI

- Prevents overlapping of further command(s)
- Example:
  - `MMEM:PACK; *WAI; MEAS? volt <PMT>`
- May be used in middle of program messages
- Must accept *WAI, extra work if you implement overlapping

Agilent Technologies

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10; PAUSE 10; PAUSE 10
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:*OPC?
read from instrument
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

Agilent Technologies

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10; PAUSE 10; PAUSE 10
stop timer
```

- [X] 0 seconds
- [ ] 10 seconds
- [ ] 20 seconds
- [ ] 30 seconds

All three commands are rapidly accepted by the instrument and placed in the input buffer before even the first is parsed. Therefore only the I/O time (probably less than 1 millisecond) is captured by the timer.

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:*OPC?
read from instrument
stop timer
```

- [X] 0 seconds
- [ ] 10 seconds
- [ ] 20 seconds
- [ ] 30 seconds

In this case since the instrument is not currently executing an overlapped command, *OPC? will instantly return its response (the number 1) so the timer should only measure the I/O time. This would typically be about 2 milliseconds.

Agilent Technologies

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10;*OPC?
read from instrument
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10; PAUSE 10; *OPC?
read from instrument
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

Agilent Technologies

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10;*OPC?
read from instrument
stop timer
```

- ☐ 0 seconds
- ☒ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

This is the conventional application of *OPC?. It does not return its response until the PAUSE 10 has completed.

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10; PAUSE 10; *OPC?
read from instrument
stop timer
```

- ☐ 0 seconds
- ☒ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

This only requires about 10 seconds because the two pause commands are overlapped and execute in parallel.

Agilent Technologies

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10; *WAI;PAUSE 10; *OPC?
read from instrument
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10;*OPC?; PAUSE 10; *OPC?
read 2 values from instrument
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☐ 20 seconds
- ☐ 30 seconds

Agilent Technologies

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10; *WAI;PAUSE 10; *OPC?
read from instrument
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☒ 20 seconds
- ☐ 30 seconds

Here, the *WAI command stops the parser until the first PAUSE is completed. Therefore the second will not execute until the first has completed. Finally, the *OPC? will not return its result until the second completes. Therefore the total time is 20 seconds.

Suppose an instrument has a command 'PAUSE'. This command has a single parameter which is the number of seconds to pause. This is an overlapped command. Approximately what time will the timer record?

```
start timer
send to instrument:PAUSE 10;*OPC?; PAUSE 10; *OPC?
read 2 values from instrument
stop timer
```

- ☐ 0 seconds
- ☐ 10 seconds
- ☒ 20 seconds
- ☐ 30 seconds

Trigger

This section describes the SCPI trigger commands and how they can be easily subsetted for instruments that do not need the full complexity.

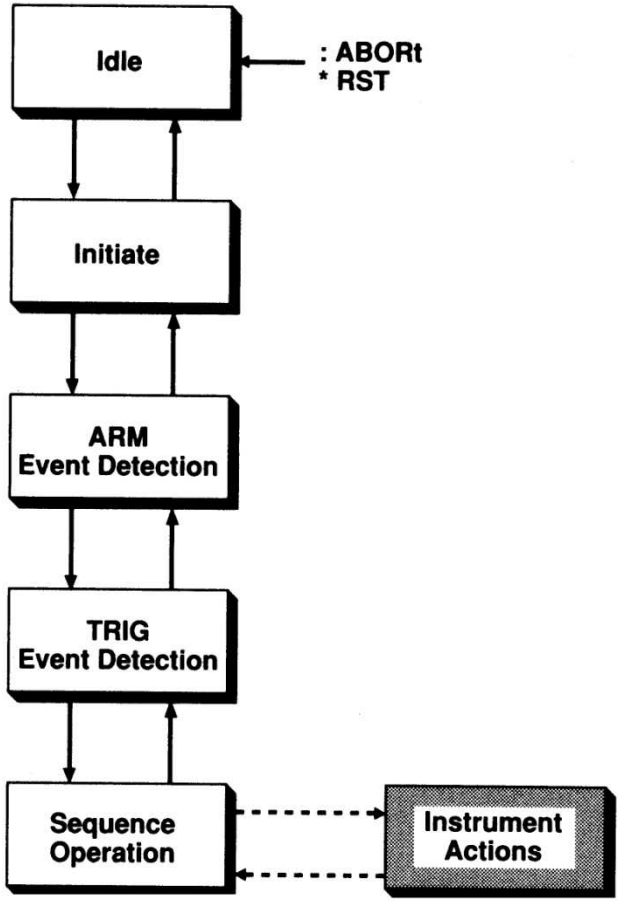
Agilent Technologies

TRIGger Subsystem - features

- Is quiescent until told otherwise
- Sets up how the instrument is to synchronize its actions with internal and external events
- Simple model meets typical needs
- Complex model allows for triggering of each sample in a vector

Agilent Technologies

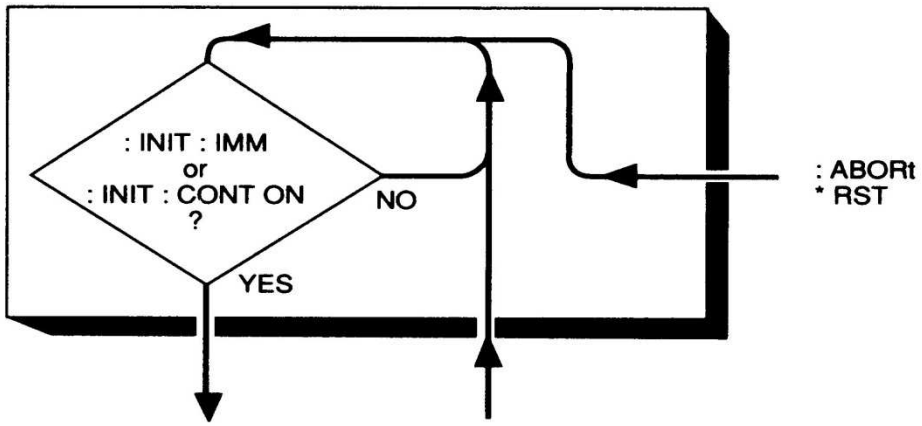
Simple trigger model



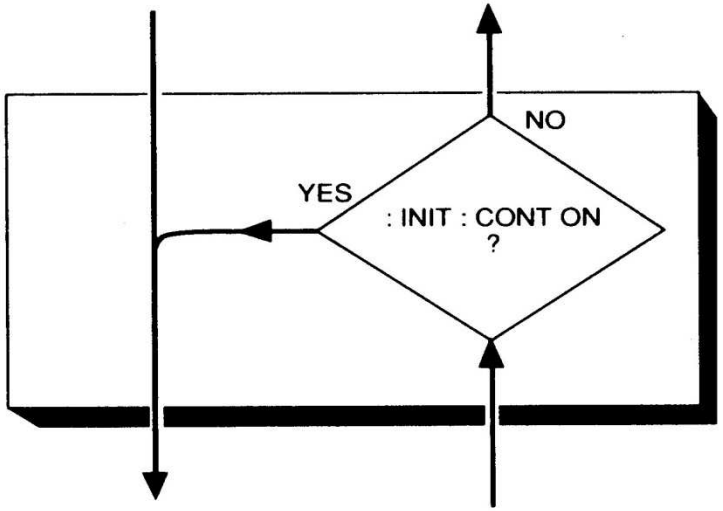
ARM-TRIG Configuration

Agilent Technologies

Trigger
- Idle and initiate states



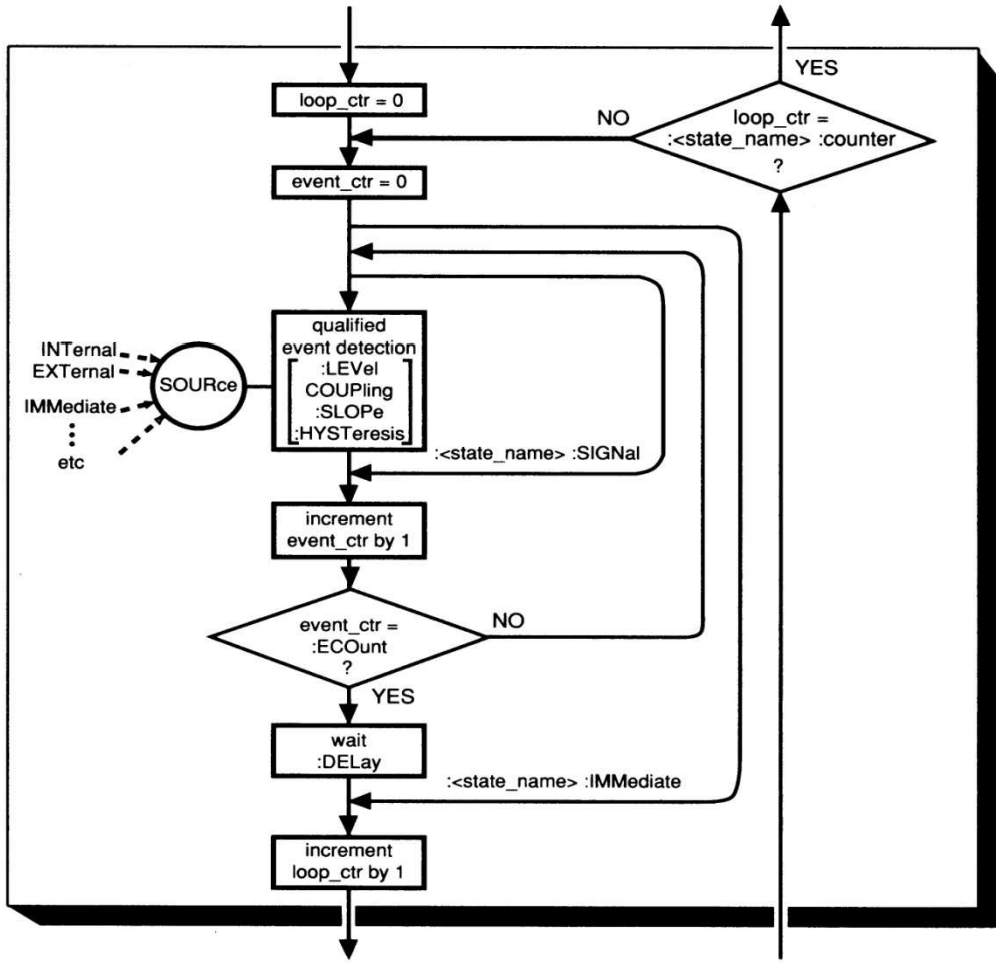
Inside the Idle State



Inside the Initiate State

Agilent Technologies

Trigger
- Event detectors

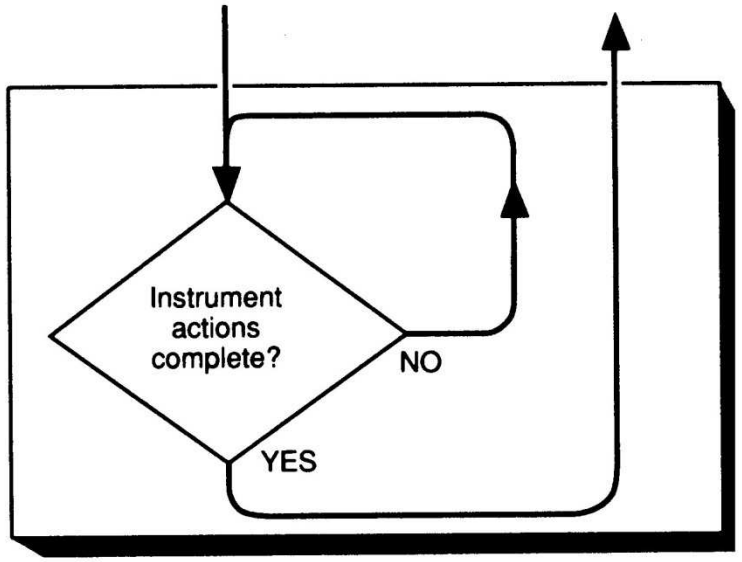


Inside an Event Detection State

Agilent Technologies

Trigger
- The device Action

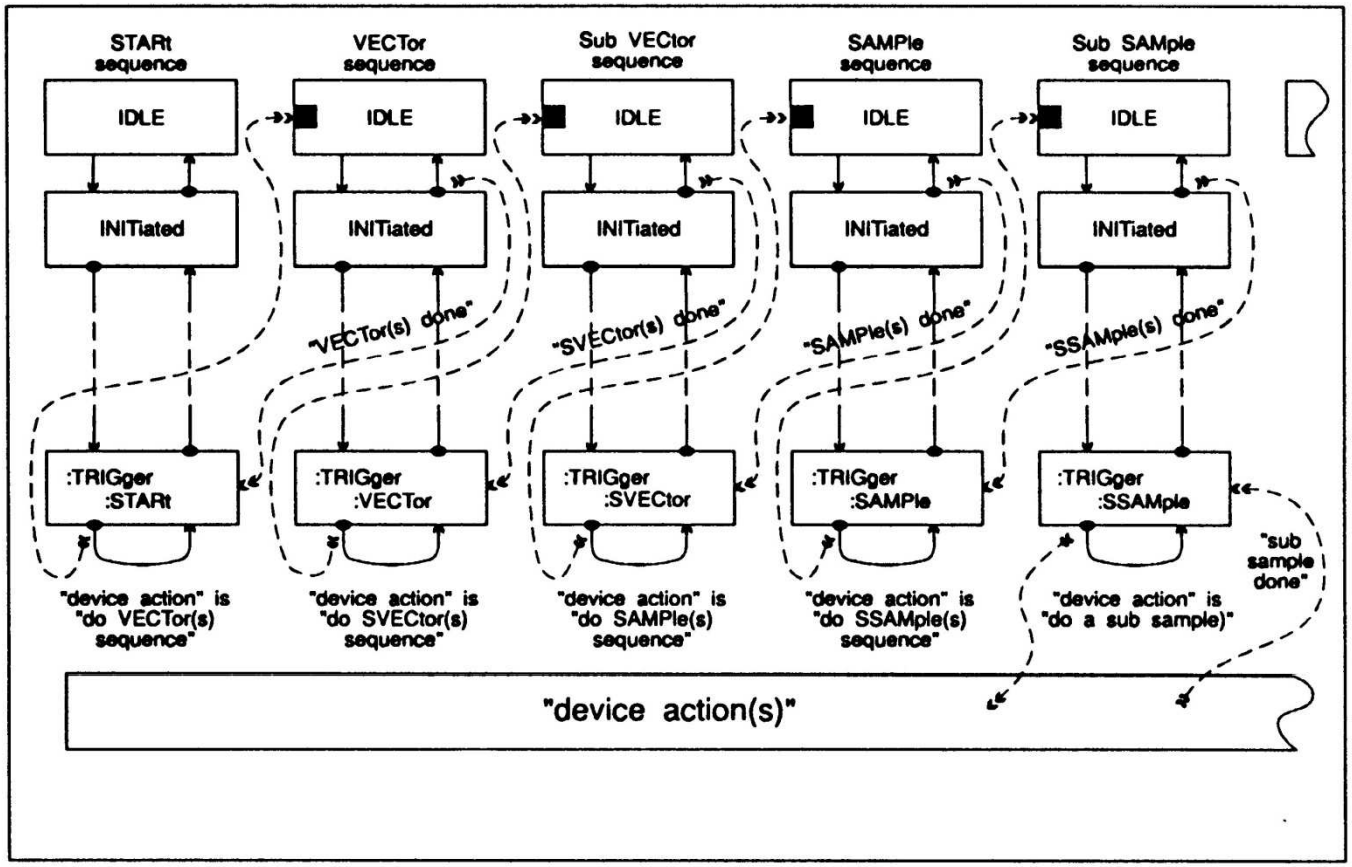
- Completion dependent upon definition of the action
  - "do a sweep" vs "start a sweep"



Inside the Sequence Operation State

Agilent Technologies

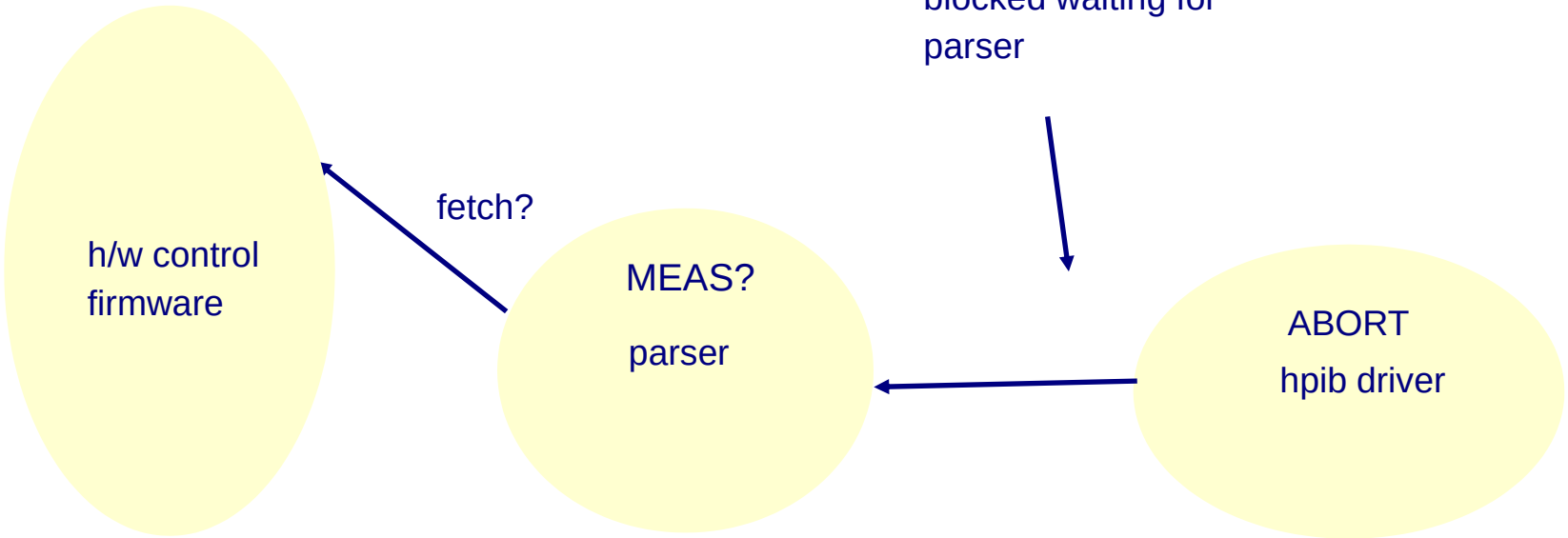
Trigger
- Any way you want it



Agilent Technologies

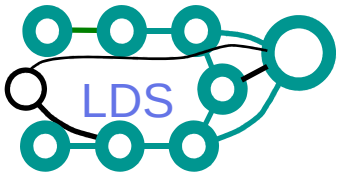
TRIGger Subsystem - ABORt

- ABORt not reliable on its own
- Must use with DCL (device clear or selected device clear)
  - MEAS? period <PMT> ABORT<PMT>



Agilent Technologies

Talking Forever



- For simple measuring device that needs to source a response whenever controller reads
- Execution routine or formatting routine continually sends responses
- Abort on interrupted (or device clear)
  - Get call out from HP-IB driver indicating interrupted
  - Don't forget device clear!

Agilent Technologies

Which of the following trigger commands are NOT found in the TRIG subsystem. That is, which of these are found at the root level of the SCPI tree?

- ☐ A) ABORt
- ☐ B) HYSTeresis
- ☐ C) INITiate
- ☐ D) A and C
- ☐ E) All of the above

Which of the following command sequences would configure an instrument to take a single measurement on external trigger then return to the idle state?
In each case assume that the various trigger state variables are all in their *RST state.

- ☐ TRIG:SOUR EXT;
- ☐ TRIG:SOURCE EXT; INITIATE
- ☐ TRIG:SOUR EXT; INITIATE:CONT ON
- ☐ TRIG:SOURCE:EXT 1;

Agilent Technologies

Which of the following trigger commands are NOT found in the TRIG subsystem. That is, which of these are found at the root level of the SCPI tree?

- [ ] A) ABORt
- [ ] B) HYSTeresis
- [ ] C) INITiate
- [X] D) A and C
- [ ] E) All of the above

Which of the following command sequences would configure an instrument to take a single measurement on external trigger then return to the idle state?
In each case assume that the various trigger state variables are all in their *RST state.

- [ ] TRIG:SOUR EXT;
- [X] TRIG:SOURCE EXT; INITIATE
- [ ] TRIG:SOUR EXT; INITIATE:CONT ON
- [ ] TRIG:SOURCE:EXT 1;

Agilent Technologies

How many measurements will be made by the following sequence:

```
TRIG:SOUR Internal
TRIG:ECOUNT 3
TRIG:COUNT 12
INITIATE
```

- ☐ 3
- ☐ 4
- ☐ 12
- ☐ 36

How many triggers are required to generate the 12 measurements made by the following sequence:

```
TRIG:SOUR External
TRIG:ECOUNT 3
TRIG:COUNT 12
INITIATE
```

- ☐ 3
- ☐ 4
- ☐ 12
- ☐ 36

Agilent Technologies

How many measurements will be made by the following sequence:

```
TRIG:SOUR Internal
TRIG:ECOUNT 3
TRIG:COUNT 12
INITIATE
```

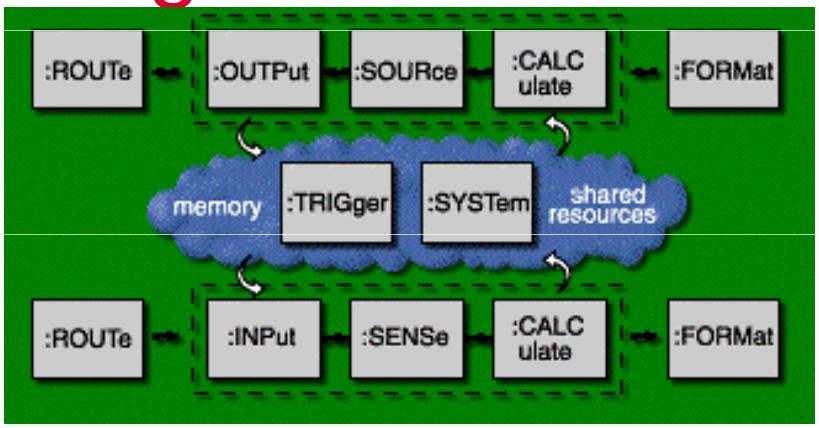
- [ ] 3
- [ ] 4
- [x] 12
- [ ] 36

How many triggers are required to generate the 12 measurements made by the following sequence:

```
TRIG:SOUR External
TRIG:ECOUNT 3
TRIG:COUNT 12
INITIATE
```

- [ ] 3
- [ ] 4
- [ ] 12
- [x] 36

System Subsystem & Status Reporting



This section provides an overview of the SCPI system subsystem, status handling, and error handling.

The SCPI System subsystem provides various housekeeping functions like control of communications.

The status subsystem starts with the 488.2 status requirements and provides a standard way of reporting other status conditions.

The SCPI error queue provides a simple mechanism for recording and reporting errors.

Agilent Technologies

Instrument Model - SYSTem Subsystem

- General system housekeeping
- Language switching
- HEADer help query
- error/event queue

Agilent Technologies

SYSTem Subsystem - error queue

- Error queue
  - FIFO
  - Read with SYSTem:ERRor?
  - When empty always returns:

    0, "No error"
  - When almost full last entry becomes:

    -350, "Queue overflow"
  - First part of string fixed by SCPI, rest for more detail

    -111, "Parameter missing;FOO:BAR<NL><Err>"

Agilent Technologies

Error numbers

- error queue number groups
  - \+ positive numbers for device designers
  - 0 no error
  - -100 syntax errors
  - -200 execution errors
  - -300 device failure
  - -350 error queue overflow
  - -400 query error
  - -500 power on
  - -600 user request
  - -700 request control
  - -800 operation complete

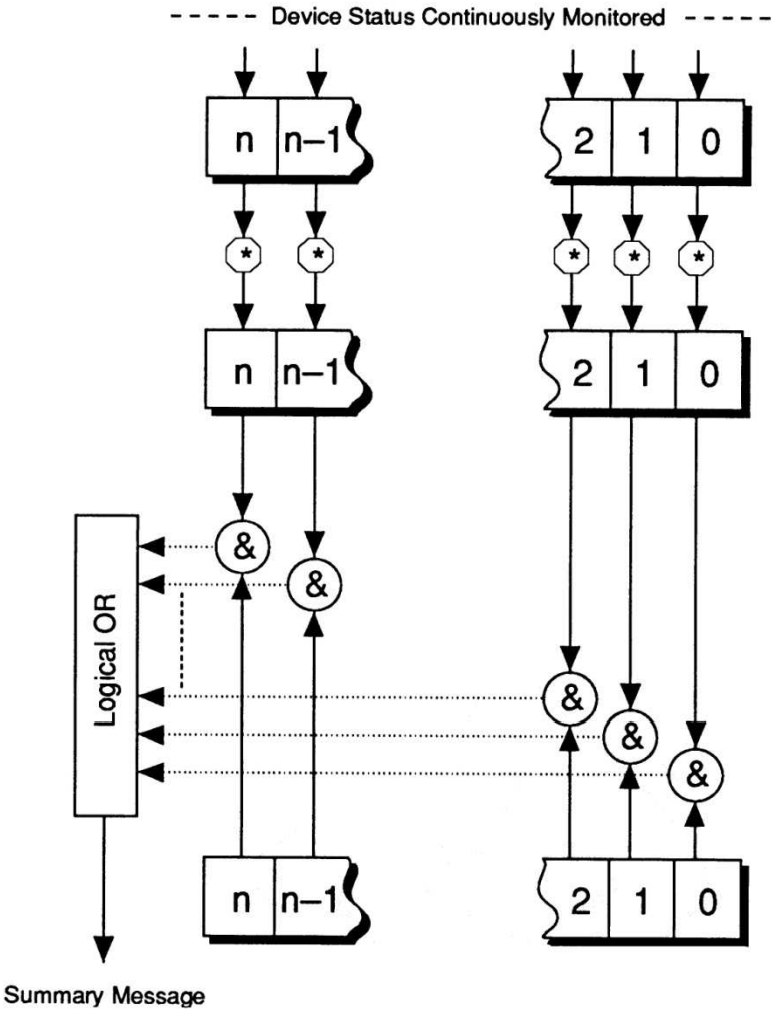
Agilent Technologies

Status reporting

- IEEE 488.1 SRQ and STB model
- IEEE 488.2 model of register operation
- SCPI naming of registers and bits

Agilent Technologies

Status reporting
- 488.2 Register model (1 of 2)



Condition register

Transition filter

Event register

Enable register

Agilent Technologies

Status reporting - 488.2 Register model (2 of 2)

- Condition register
  - Dynamic indication of current status
- Transition filter
  - Determines what to "sample and hold" in event register
  - Always edge triggered
  - Select: rising, trailing or both edges
- Event register
  - Buffered condition register (based on transition filter)
- Enable register
  - Selects bits to OR into summary

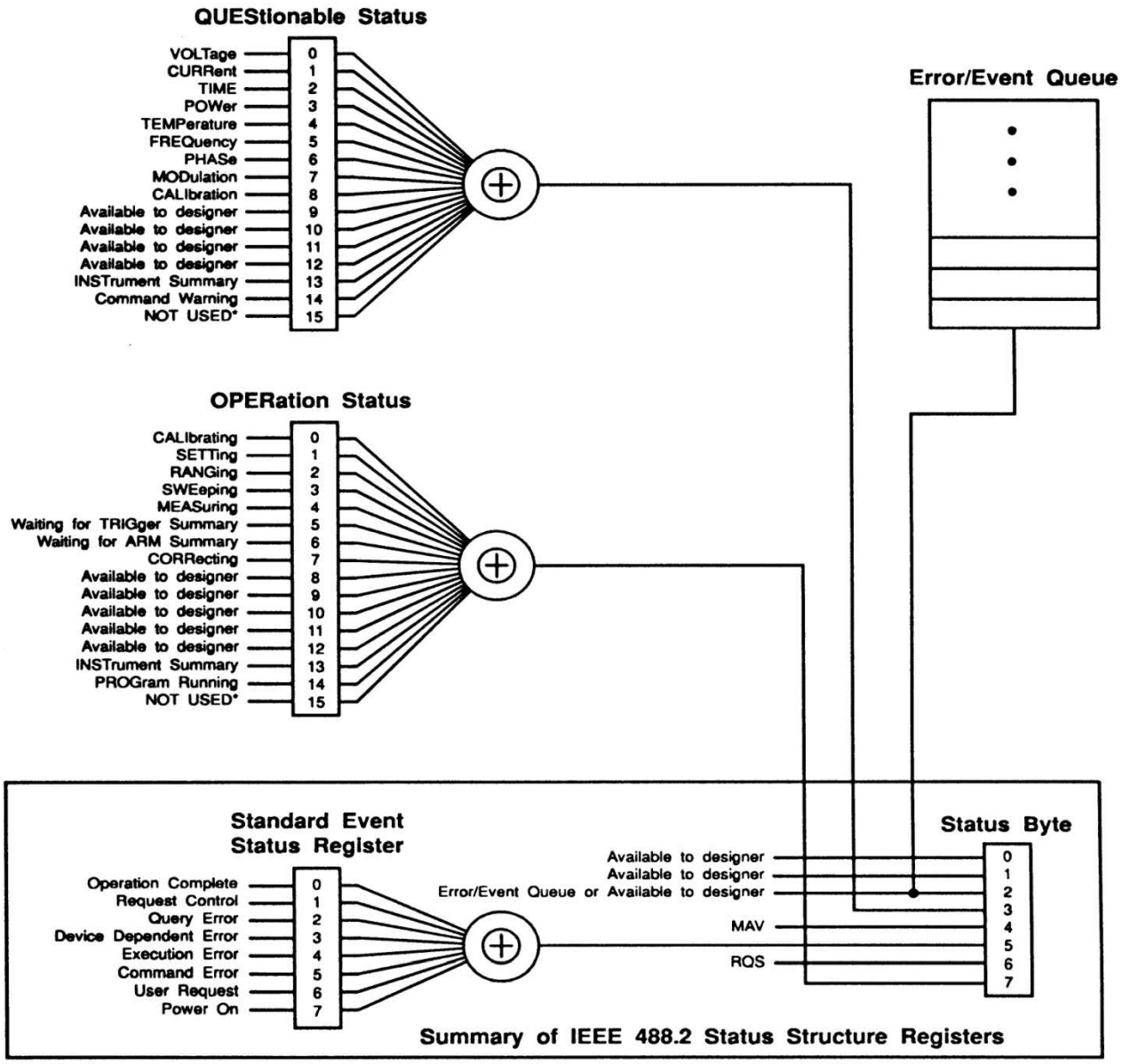
Agilent Technologies

Status Reporting
- 488.2 power on status bit

- registers unaffected by *RST
- power on status control *PSC flag effects retention of status information
  - see STATus subsystem

Agilent Technologies

Status reporting - Summary



Agilent Technologies

Known state
- recommended sequence

```
DCL|SDC   get attention of parser
*RST              state vars to known
values
*CLS              clear 488.2 status
reg's
*SRE 0            clear SRQ enable
mask
*ESE 0            clear std. event
enable
STATUS:PRESET
          clean up other registers

start your programming
```

Agilent Technologies

Which of the following are valid responses to SYST:ERR?

- ☐ 0,"No Error"
- ☐ -420, "Query Error"
- ☐ -213, "Syntax Error; Space require after header"
- ☐ -213, "Syntax Error; Space required after header:SENSE:VOLT^12"
- ☐ All of the above

What is the difference between executing a serial poll and sending the instrument "*STB?" and reading the response?

- ☐ They are exactly the same
- ☐ They differ because "*STB?" must be parsed and therefore cannot be used to interrogate the instrument while it is parsing or executing other commands. For instance, it is meaningless to check "message available" with "*STB?" because the query itself produces a response.

Agilent Technologies

Which of the following are valid responses to SYST:ERR?

- ☐ 0,"No Error"
- ☐ -420, "Query Error"
- ☐ -213, "Syntax Error; Space require after header"
- ☐ -213, "Syntax Error; Space required after header:SENSE:VOLT^12"
- ☒ All of the above

What is the difference between executing a serial poll and sending the instrument "*STB?" and reading the response?

- ☐ They are exactly the same
- ☒ They differ because "*STB?" must be parsed and therefore cannot be used to interrogate the instrument while it is parsing or executing other commands. For instance, it is meaningless to check "message available" with "*STB?" because the query itself produces a response.

Agilent Technologies

Which of the following enable an interrupt on message available?

- ☐ *SESE 16; *SRE 32
- ☐ *SRE 16
- ☐ *STB 16
- ☐ SYSTEM:ENABLE:RQS:MAV

Which of the following enable an interrupt on operation complete. That is, an interrupt will be generated when the *OPC command determines that overlapped command execution has completed.

- ☐ *SESE 1; *SRE 32
- ☐ *SRE 16
- ☐ ~~*OPC:STATE ON~~
- ☐ *SRE 254; *SRE 223

Agilent Technologies

Which of the following enable an interrupt on message available?

- [ ] *SESE 16; *SRE 32
- [x] *SRE 16
- [ ] *STB 16
- [ ] SYSTEM:ENABLE:RQS:MAV

Which of the following enable an interrupt on operation complete. That is, an interrupt will be generated when the *OPC command determines that overlapped command execution has completed.

- [x] *SESE 1; *SRE 32
- [ ] *SRE 16
- [ ] ~~*OPC:STATE ON~~
- [ ] *SRE 254; *SRE 223

Duplicate Functionality & Display

There are many cases in instrument design where there are multiple instances of similar or identical functions. One common example of this is multiple input channels, another common example is multiple windows on a display.

This section describes how SCPI deals with this duplicated functionality and the FEED command that is used in configuring these complex instruments. It also describes the display system that uses this capability.

Agilent Technologies

Multiple channels - multiple parts

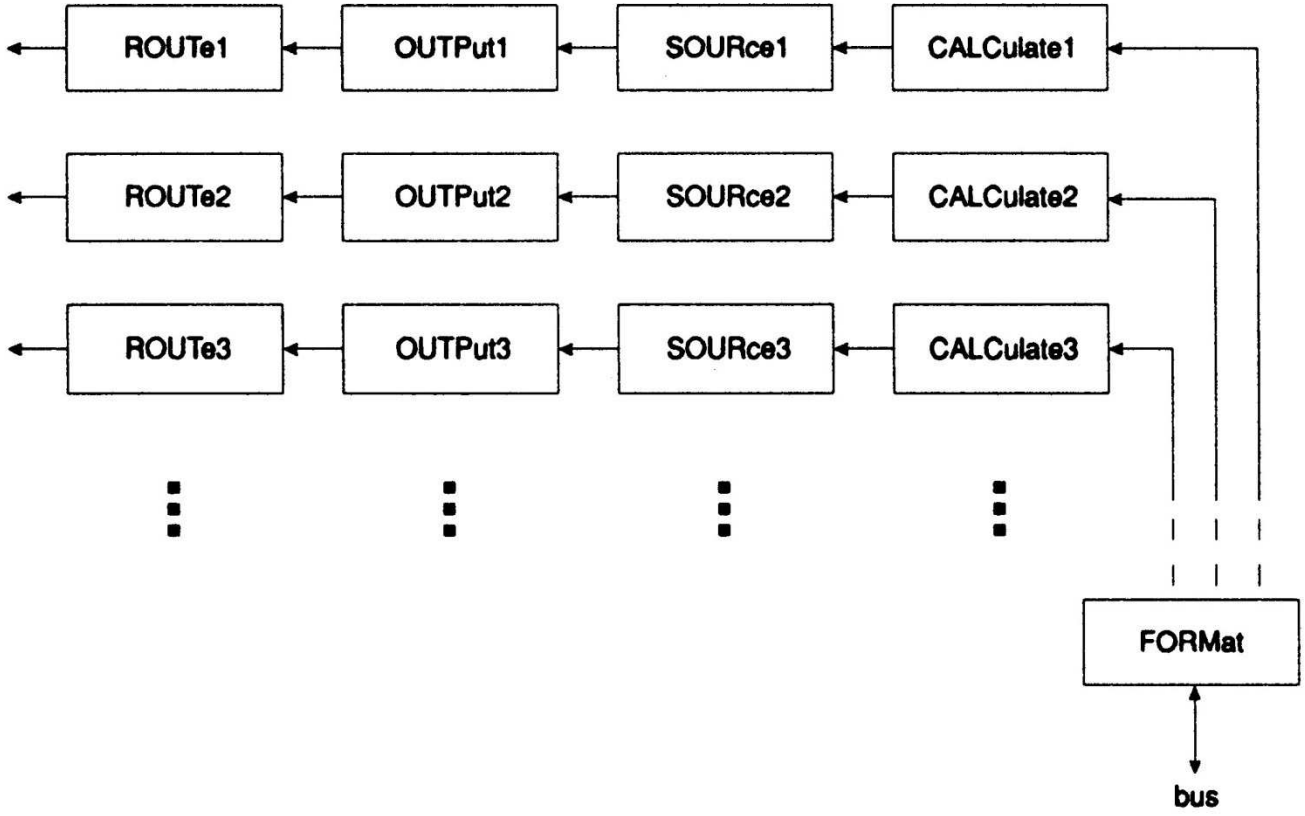
- Multiple independent channels
  - The model is cloned as needed
  - Each clone has a different instance number
  - This number is appended to the mnemonic, called:

    Header suffixes

    Sometimes sub-opcodes
- Good coverage in the 1st chapter of SCPI Command Reference

Agilent Technologies

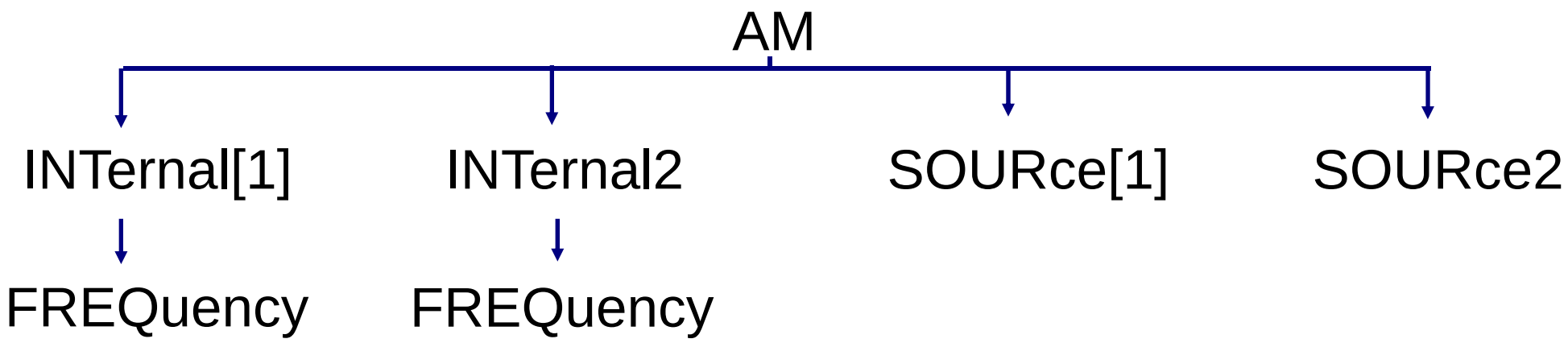
Cloned model
- for multiple identical functionality



Agilent Technologies

Multi-channel programming example

- Subsystems may also be cloned



Example: Set up the second audio source

✗ `AM:SOUR2 internal2; INT:FREQ 2khz <PMT>`

✓ `AM:SOUR2 internal2; INT2:FREQ 2khz <PMT>`

Agilent Technologies

Programming with sub-opcodes

- Watch out for mnemonic and suffix combinations > 12 characters
- Don't send suffix of 1
  - Program written using a single channel for box that supports suffixes, will not work with simple box that has no suffixes
  - Only need to implement suffixes on those parts of your tree that need them

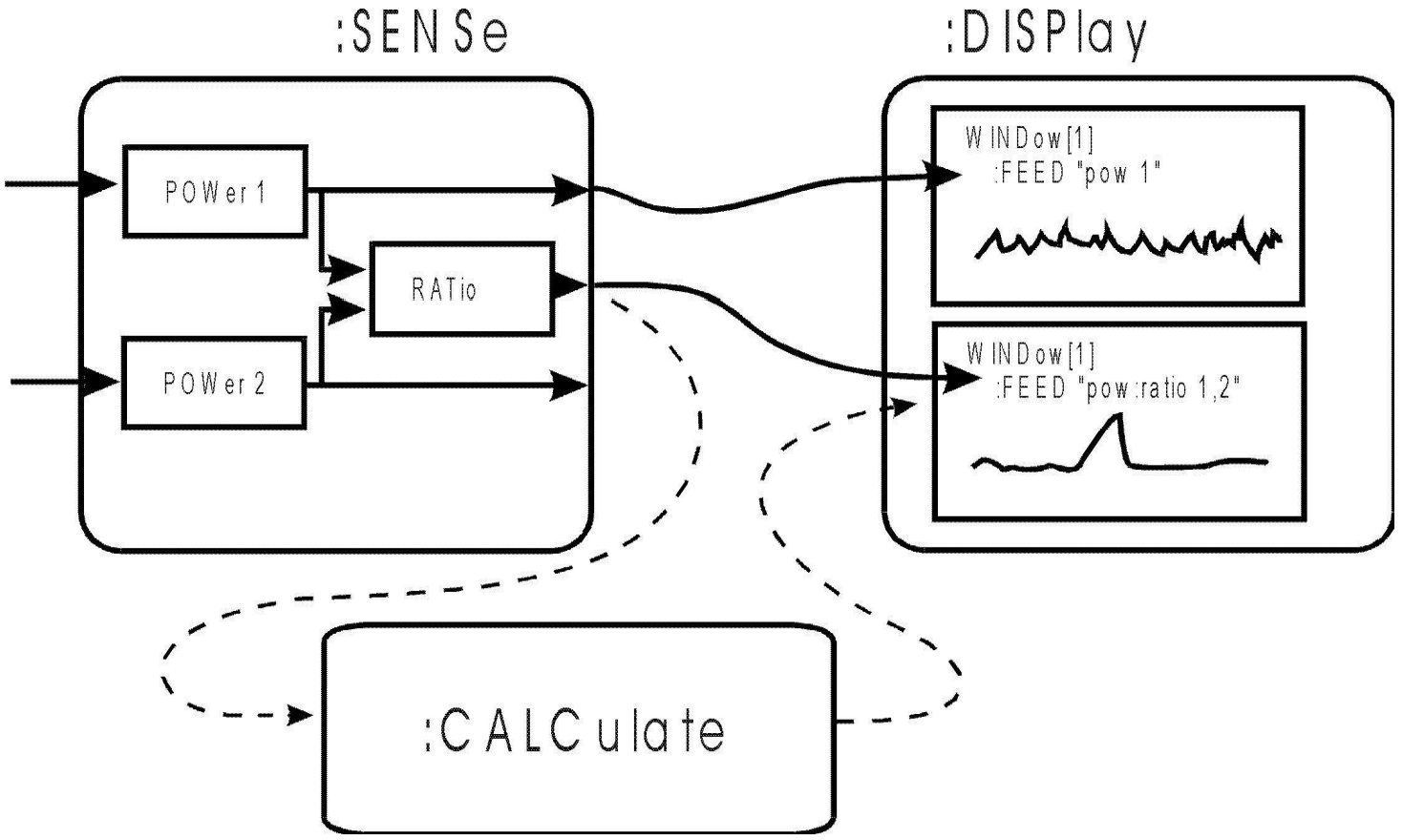
Agilent Technologies

...:FEED command

- AM:SOURCE is example of (early) dedicated feed command
- FEED affords opportunity to pick "data" up from different points around the instrument
- Data is used in an abstract sense
  - It may be a signal
  - It may be converted data
- Implementation controls which stream a given FEED may select
- Note analogous event feed in TRIGger

Agilent Technologies

DISPlay:FEED



Agilent Technologies

DISPLAY subsystem

- In simple case
  - 1 DISPlay
  - 1 WINDow - which is a "default node" effectively transparent to user
- Has simple X, Y, and R controls
- Data FEED defaults to CALCulate[1]

Agilent Technologies

Simple DISPlay tree

```
DISPlay
 [:STATe]
  :ANNotation[:ALL]
  :BRIGhtness
 [:WINDow]
    :X
      :LABel
     [:SCALe]
        :AUTO
        :LEFT
        :RIGHt
    :Y
       ...
```

Agilent Technologies

For an instrument that has a two-window display, how would you set the label on the y axis of the second window to be "MyAxis"?

- ☐ DISPLAY:WINDOW2:Y:LABEL "MyAxis"
- ☐ DISPLAY2:Y:LABEL "MyAxis"
- ☐ DISPLAY:Y:WINDOW2:LABEL "MyAxis"
- ☐ DISPLAY:WINDOW:Y:FEED "LABEL:""MyAxis"""

Agilent Technologies

For an instrument that has a two-window display, how would you set the label on the y axis of the second window to be "MyAxis"?

- [x] DISPLAY:WINDOW2:Y:LABEL "MyAxis"
- [ ] DISPLAY2:Y:LABEL "MyAxis"
- [ ] DISPLAY:Y:WINDOW2:LABEL "MyAxis"
- [ ] DISPLAY:WINDOW:Y:FEED "LABEL:""MyAxis"""