



Turn-on, Tune-in, Tape-out

Extracting register details from
documents using Spec2Reg

Dr. Ambar Sarkar

Chief Verification Technologist

Paradigm Works Inc.



Introduction – Distinguished Pedigree



- Paradigm Works was founded in September 2000
- Michael Hoyt CEO and Jim Crocker VP Engineering
- Headquartered in Andover, Massachusetts

Background – ASIC and FPGA Experts



- Reputation for excellence and integrity
- Highest value proposition in industry
 - High quality engineering resources combined with low cost structure
- SW Products to support system design and verification - ReleaseWorks™ (GVP), FrameWorks™, Verification IP (PCI Express, Ethernet)
- Blue Chip Client List
- Global Relationships

Purpose - Excellence in R&D



- To provide the most effective ASIC and FPGA technology and development services in the industry
- Bring our systems understanding to the development of increasingly complex ASIC's and FPGA's
- Long term mutually successful relationships



Paradigm Works - Clients





The Problem

- Register class creation not fully automated
- Difficult to go from design specification to final output
- Consistency between specification and implementation
- Fixed formats



The Problem

- Home grown solutions exist but...
 - Usually concoction of scripts
 - Maintenance issues
 - Takes time away from verification
 - And no one really addresses the extraction of the details from the “source” doc
 - Word/Pdf etc

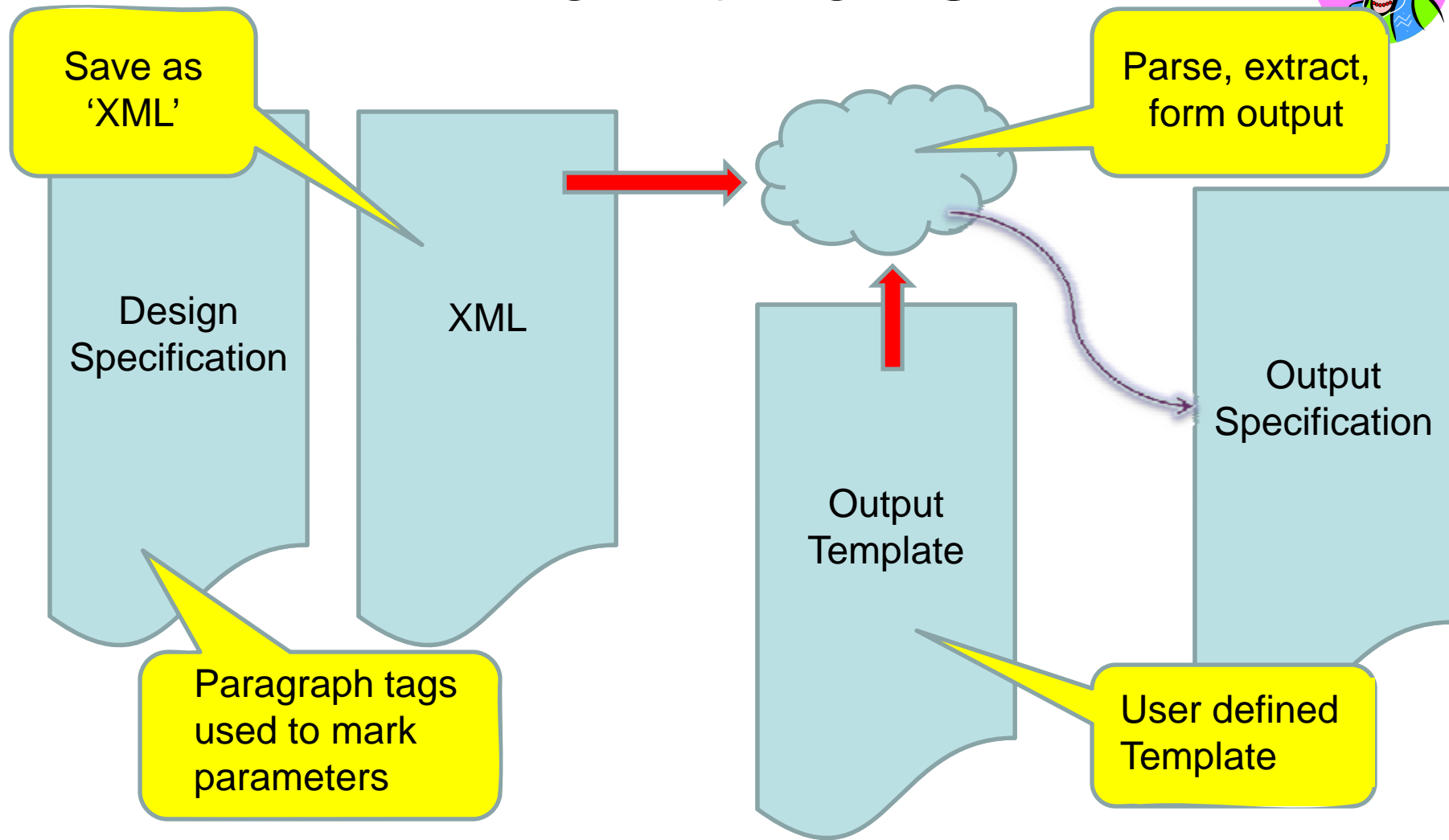


How it can be addressed

- Divide the problem
- Define independent mechanisms that:
 - Extract register parameters *from* the design specification
 - Convert the parameters *to* the final output
 - RAL



How it works





Example Input Tagging

Register Name	CONTROL			
Register Address	<i>0x12345678</i>			
Field name	Attribute	Bit position	Reset	Description
ENABLE	RW	7	0	
MODE	RW	6:0	0:16	
SYNC	RO			
START	RW			

Paragraph tag is 'regname'

Paragraph tag is 'regaddr'

Paragraph tag is 'regfield'

Paragraph tag is 'regattr'

Paragraph tag is 'regfieldpos'

Paragraph tag is 'regreset'



Example Output Template

```
[% FOREACH register IN the_registers -%]  
register [% register.name %] {  
    bytes [% register.size %];  
  
[% IF register.ftype == "register" -%]  
    field [% register.fname %]  
        {bits [% register.num_bits %];  
         access [% register.attr %];  
         reset [% register.reset %]}  
}  
}
```

Extracted from
'regname' tag in
the design
specification

Extracted from
'regfield' tag in
the design
specification

Extracted from
'regfieldpos' tag
in the design
specification



Example Output

```
register FifoErr2MaskReg @'h11 {
  bytes 1;
  left_to_right;

  field OVERFLOW_ERR      {bits 4; access rw; reset 0x0}
  field UNDERFLOW_ERR    {bits 4; access rw; reset 0x0}
}
register Fifo1OvrFlwCntReg @'h14 {
  bytes 1;
  left_to_right;

  field FIFO_OVR_CNT      {bits 8; access rc; reset 8'h00}
}
register Fifo2OvrFlwCntReg @'h15 {
  bytes 1;
  left_to_right;
  . . .
```



Groovy?

- 'Turn on' meant go within to activate your neural and genetic equipment
 - Be smart.. Go to the source specification
- 'Tune in' meant interact harmoniously with the world around you
 - Work seamlessly the existing VMM application RAL
 - Share the tool in open source (SourceForge)



Groovy?

- 'Drop Out' meant self-reliance, a discovery of one's singularity, a commitment to mobility, choice, and change.
 - Hopefully a **tape-out** will lead to all these!

