Implementation of a Computer Vision project on multiple platforms

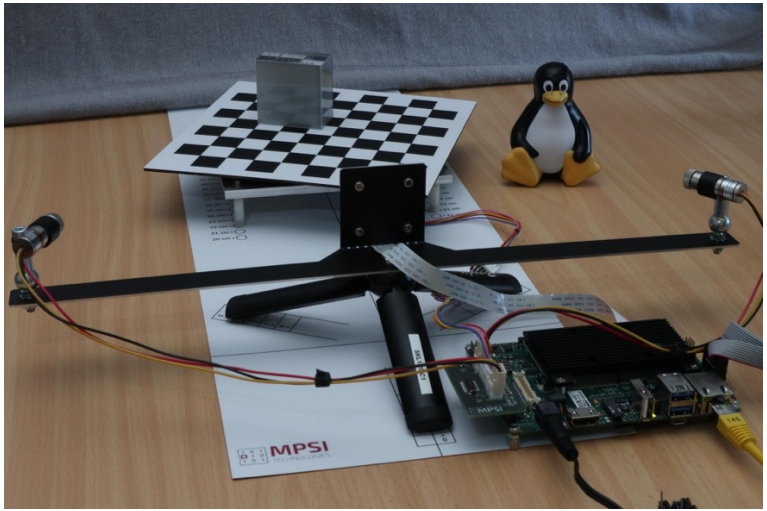
MPSI TECHNOLOGIES

Alexander Wirthmüller
aw@mpsitechnologies.com

Multiple Platforms

Why | Which | How

- Being able to switch platform is attractive
  - Evolved landscape of offerings
  - Best match in technology and price point may be with another vendor
  - (Supply chain issues)
- Specific motivation here: Whiznium Computer Vision (CV) Demonstrator



Multiple Platforms

Why | Which | How

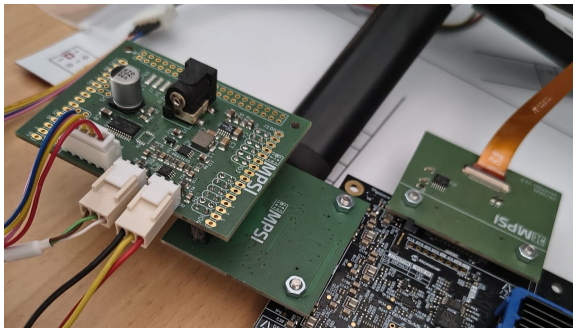
- Comprehensive sample of the current mid-range

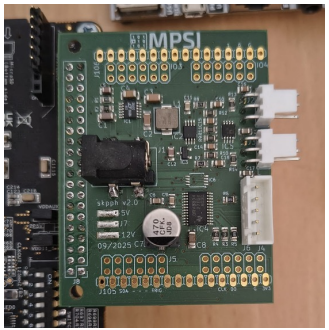
	Platform and board	Year (node)	CPU	MIPI	DDR memory	extra	Build system and kernel	Vendor tool
ag	Altera Agilex 3 Agilex 3 SoC Dev kit	2025 ("7" nm)	ARMv8	hybrid IP	hard IP PL+ PS shared		Yocto 5.3 6.18	Quartus Prime 26.1
nx	AMD Artix 7 Digilent Nexys Video	2011 (28 nm)	-	LVDS (ext. pass.) limited	I/O	PCIe	(ubuntu host)	Vivado 2024.2
ay	AMD Zynq 7020 Digilent Arty Z7-20	2011 (28 nm)	ARMv7	LVDS (ext. pass.) limited	hard IP PS shared		Yocto 5.0.4 6.6.40	Vivado 2024.2
zu	AMD Zynq MPSoC 1CG Avnet ZUBoard	2015 (16 nm)	ARMv8	I/O	hard IP PS shared		Yocto 5.0.4 6.6.40	Vivado 2024.2
ti	Efinix Titanium Ti180 Dev kit	2021 (16 nm)	soft RISC-V	hard IP	hard IP		Buildroot 2021.1 5.10	Efinity 2025.1
cl	Lattice Crosslink-NX Eval board	2020 (28 nm)	-	hard IP	-	PCIe	(ubuntu host)	Radiant 2025.2
dc	Microchip Polarfire SoC Disco kit	2019 (28 nm)	RISC-V	I/O limited	hard IP PS shared		Yocto 5.0.16 6.12.22	Libero SoC 2025.1

Multiple Platforms

Why | Which | How

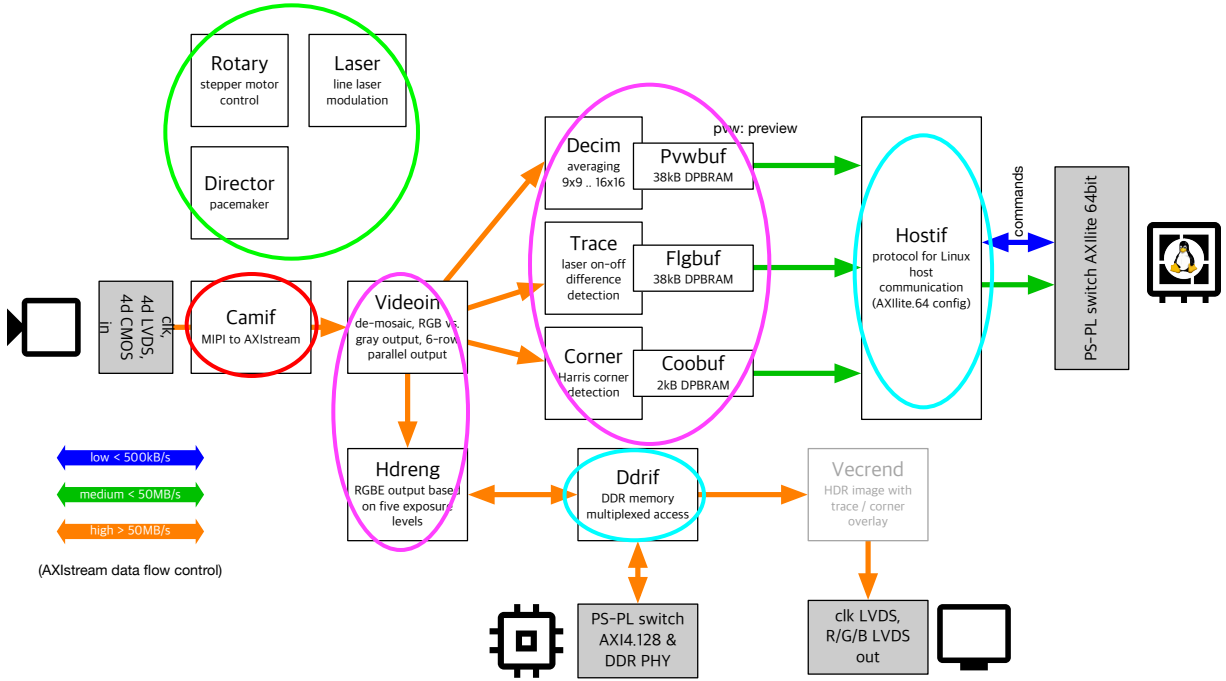
- Timeline / sequence: zu --(new vendor)---> ti --(new vendor)---> dc --(older generation)---> ay --(non SoC)---> nx --(new vendor)---> cl --(new vendor)---> ag
- Use free vendor and simulation tools only
- Universal peripheral PCB with PMOD/RPi connectors, QSE adapter, FMC for PCIe





Design

Overview | Video pipeline



- Invariant
  - Low-speed peripherals
  - Video pipeline (?!)
- Whiznium-invariant
  - Host interface
  - DDR interface
- Manual labor
  - High-speed peripherals

- Videoin: de-mosaic from (dual) RAW12 to RGB and 6-row gray12; sophisticated pixel-packing
- Decim: preview image generation with adjustable edge parameter (e.g. 7x7 or 10x10); multiplication + shift to approximate division
- Corner: pipelined Harris corner detection algorithm; score calculation with multipliers then maxfind
- Trace: line laser on-off difference image determination exp5 intermediate result format
- Hdreng: find RGBE representation (E = exponent) for sequence of five different exposure times; leverages DDR memory

RTL Journey

MIPI, DDR & PCIe | Block RAM | Simulation | Timing closure | Tooling quirks

- MIPI

  Altera MIPI DPHY IP 6.2.0 + MIPI CSI-2 IP 3.1.0
  AMD MIPI CSI-2 Receiver Subsystem (cf. PG232)
  Efinix MIPI 2.5G CSI-2 RX Controller Core / hardened
  Lattice MIPI CSI/DSI IP (cf. IPUG-02321) / hardened
  Microchip IOD 1.0.0 + MIPI CSI-2 Receiver Decoder IP 5.1

- DDR memory

  Altera (hardened controller via SoC AXI4 interconnect)
  AMD Memory Interface Generator 4.2 (cf. UG586)
  Efinix (hardened controller via AXI4)
  Microchip (hardened controller via SoC AXI4 interconnect)

- PCI Express

  AMD Integrated Block for PCI Express 3.3 (cf. UG477)
  Lattice PCIe X1 IP Core 1.1.2 (cf. IPUG-02091)

RTL Journey

MIPI, DDR & PCIe | Block RAM | Simulation | Timing closure | Tooling quirks

Memory component	ag	ay (host/32)	cl (host/8)	dc	nx	ti (host/32)	zu	Rationale
corner.coobuf	2 kB 32/64	-	2 kB 32/8	2 kB 32/64	2 kB 32/8	2 kB 32/32	2 kB 32/64	512x row16:col16
corner.imd0..4buf	2 kB 8/8	-	2 kB 8/8	2 kB 8/8	2 kB 8/8	2 kB 8/8	2 kB 8/8	2000+ pix8
decim.accbuf	2 kB 64/64	2 kB 64/64	2 kB 64/64	2 kB 64/64	2 kB 64/64	2 kB 64/64	2 kB 64/64	200+ rgb64
decim.pvwbuf	38 kB 8/64	38 kB 8/32	38 kB 8/8	38 kB 8/64	38 kB 8/8	32 kB 32/8	38 kB 8/64	1296x972 edge10 rgb24
hdreng.rowA/Bbuf	8 kB 128/128	-	-	8 kB 32/64	**8 kB 32/128**	**8 kB 32/128**	8 kB 32/128	2000+ rgbe32
trace.accbuf	188 kB 8/8	-	3x 64 kB 8/8	188 kB 8/8	188 kB 8/8	12x 16 kB 8/8	188 kB 8/8	640x480 2log5
trace.flgbuf	38 kB 8/64	-	38 kB 8/8	38 kB 8/64	38 kB 8/8	32 kB 8/32	38 kB 8/64	640x480 pix1
videoin.evenA/Bbuf, oddbuf	6 kB 64/64	6 kB 64/64	6 kB 64/64	6 kB 64/64	6 kB 64/64	8 kB 64/64	6 kB 64/64	4000+ raw12, some with different clocks
videoin.row0..4buf	4 kB 64/64	4 kB 64/64	4 kB 64/64	4 kB 64/64	4 kB 64/64	4 kB 64/64	4 kB 64/64	2000+ gray12

bold: not aWr;bRd
underline: different clocks

- wWrite == wRead: adapt writing
- registered input / output: adapt state machines
- smaller buffer size: add parametrization / ROI within frame

RTL Journey

MIPI, DDR & PCIe | Block RAM | Simulation | Timing closure | Tooling quirks

- Core video pipeline in Cocotb + Questa (courtesy Altera)
- Efinix only vendor not to bundle simulator
- Combination of example designs + core video pipeline sim -> success

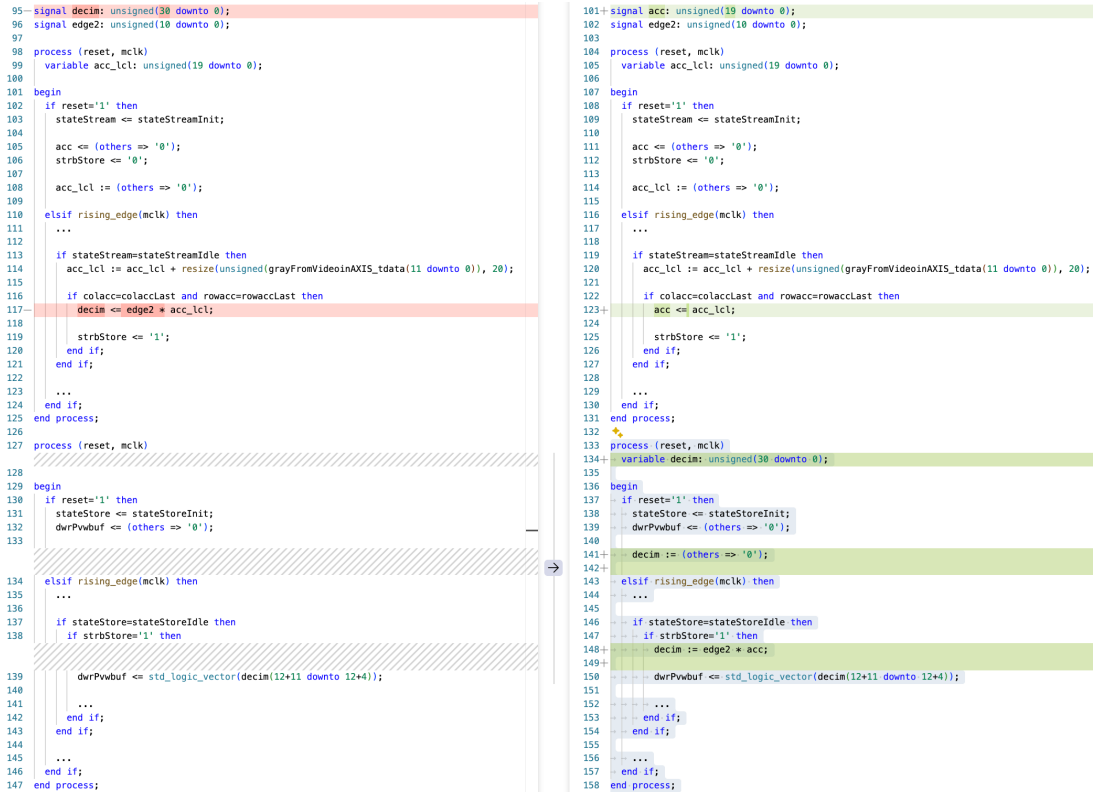
- Mostly generational divide
  - effortless master clock (mclk) == pixel clock (pixclk) at 200 MHz for latest generation (ag, ti, zu)
  - old generation (ay, cl, dc, nx): reduction of mclk to 100 MHz with clock domain crossing pixclk (coming from MIPI) to mclk in videoin; not very problematic as clear ingress / egress divide with two-port block RAM

- Departure from FSM-state combinatorial assignments

```
2  rgbAXIS_tlast <= '0' when stateEgr=stateEgrRun and aBuf/=aBufLast else '1';
3
4  process (resetPixclk, pixclk)
5
6  begin
7    if resetPixclk='1' then
8      stateEgr <= stateEgrInit;
9      aBuf <= 0;
10
11   elsif rising_edge(pixclk) then
12     if stateEgr=stateEgrInit or rng_pixclk='0' then
13       aBuf <= 0;
14
15       ...
16
17     elsif stateEgr=stateEgrIdle then
18       ...
19
20       stateEgr <= stateEgrRun;
21
22     elsif stateEgr=stateEgrRun then
23       aBuf <= aBuf + 1;
24
25       ...
26     end if;
27   end if;
28 end process;
```

```
2  process (resetPixclk, pixclk)
3
4  begin
5    if resetPixclk='1' then
6      stateEgr <= stateEgrInit;
7      aBuf <= 0;
8+     rgbAXIS_tlast <= '1';
9
10   elsif rising_edge(pixclk) then
11     if stateEgr=stateEgrInit or rng_pixclk='0' then
12       aBuf <= 0;
13+      rgbAXIS_tlast <= '1';
14
15       ...
16
17     elsif stateEgr=stateEgrIdle then
18       ...
19+
20+      rgbAXIS_tlast <= '0';
21
22       stateEgr <= stateEgrRun;
23
24     elsif stateEgr=stateEgrRun then
25       aBuf <= aBuf + 1;
26+
27+      if aBuf=aBufLast-1 then -- aBuf increment post comparison
28+        rgbAXIS_tlast <= '1';
29+      end if;
30
31       ...
32     end if;
33   end if;
34 end process;
```

- Resource-neutral example



```
signal decim: unsigned(30 downto 0);
signal edge2: unsigned(10 downto 0);

process (reset, mclk)
  variable acc_lcl: unsigned(19 downto 0);

begin
  if reset='1' then
    stateStream <= stateStreamInit;

    acc <= (others => '0');
    strbStore <= '0';

    acc_lcl := (others => '0');

  elsif rising_edge(mclk) then
    ...

    if stateStream=stateStreamIdle then
      acc_lcl := acc_lcl + resize(unsigned(grayFromVideoinAXIS_tdata(11 downto 0)), 20);

      if colacc=colaccLast and rowacc=rowaccLast then
        decim <= edge2 * acc_lcl;

        strbStore <= '1';
      end if;
    end if;

    ...
  end if;
end process;

process (reset, mclk)

begin
  if reset='1' then
    stateStore <= stateStoreInit;
    dwrPvwbuf <= (others => '0');


  elsif rising_edge(mclk) then
    ...

    if stateStore=stateStoreIdle then
      if strbStore='1' then


        dwrPvwbuf <= std_logic_vector(decim(12+11 downto 12+4));

        ...
      end if;
    end if;

    ...
  end if;
end process;
```

```
signal acc: unsigned(19 downto 0);
signal edge2: unsigned(10 downto 0);

process (reset, mclk)
  variable acc_lcl: unsigned(19 downto 0);

begin
  if reset='1' then
    stateStream <= stateStreamInit;

    acc <= (others => '0');
    strbStore <= '0';

    acc_lcl := (others => '0');

  elsif rising_edge(mclk) then
    ...

    if stateStream=stateStreamIdle then
      acc_lcl := acc_lcl + resize(unsigned(grayFromVideoinAXIS_tdata(11 downto 0)), 20);

      if colacc=colaccLast and rowacc=rowaccLast then
        acc <= acc_lcl;

        strbStore <= '1';
      end if;
    end if;

    ...
  end if;
end process;
process (reset, mclk)
  variable decim: unsigned(30 downto 0);

begin
  if reset='1' then
    stateStore <= stateStoreInit;
    dwrPvwbuf <= (others => '0');

    decim := (others => '0');

  elsif rising_edge(mclk) then
    ...

    if stateStore=stateStoreIdle then
      if strbStore='1' then
        decim := edge2 * acc;

        dwrPvwbuf <= std_logic_vector(decim(12+11 downto 12+4));

        ...
      end if;
    end if;

    ...
  end if;
end process;
```

- Elaboration order for constant (👍 Efinity)

```
32  constant wD: natural := 8;
33
34  signal txAXIS_tdata: std_logic_vector(wD-1 downto 0);
35
36  ...
37
38  process (reset, mclk)
39
40  begin
41    if reset='1' then
42      ...
43
44    elsif rising_edge(mclk) then
45      ...
46
47      if wD=8 then
48        if wordcnt=wordcntLast-1 then
49          txAXIS_tdata <= crc(15 downto 8);
50        elsif wordcnt=wordcntLast then
51          txAXIS_tdata <= crc(7 downto 0);
52        end if;
53
54      else
55        txAXIS_tdata(7 downto 0) <= crc(15 downto 8);
56        txAXIS_tdata(15 downto 8) <= crc(7 downto 0);
57      end if;
58
59      ...
60    end if;
61  end process;
```

```
38  constant wD: natural := 8;
39
40  signal txAXIS_tdata: std_logic_vector(16*8-1 downto 0);
41
42  ...
43
44  process (reset, mclk)
45
46  begin
47    if reset='1' then
48      ...
49
50    elsif rising_edge(mclk) then
51      ...
52
53      if wD=8 then
54        if wordcnt=wordcntLast-1 then
55          txAXIS_tdata(7 downto 0) <= crc(15 downto 8);
56        elsif wordcnt=wordcntLast then
57          txAXIS_tdata(7 downto 0) <= crc(7 downto 0);
58        end if;
59
60      else
61        txAXIS_tdata(7 downto 0) <= crc(15 downto 8);
62        txAXIS_tdata(15 downto 8) <= crc(7 downto 0);
63      end if;
64
65      ...
66    end if;
67  end process;
```

- std_logic_vector(13 downto var) (👍 Vivado)

```
65  process (reset, mclk)
66    variable lsb: natural range 0 to 2;
67
68  begin
69    if reset='1' then
70      lsb := 0; -- can also be 1 or 2
71
72    elsif rising_edge(mclk) then
73      ...
74
75      if ccwNotCw='0' then
76        if angle(13 downto lsb)=anglemax_vec(13 downto lsb) then
77          angle(13 downto lsb) <= (others => '0');
78        else
79          angle(13 downto lsb) <= angle(13 downto lsb) + 1;
80        end if;
81      else
82        if angle(13 downto lsb)=0 then
83          angle(13 downto lsb) <= anglemax_vec(13 downto lsb);
84        else
85          angle(13 downto lsb) <= angle(13 downto lsb) - 1;
86        end if;
87      end if;
88
89      ...
90    end if;
91  end process;
```

```
71  process (reset, mclk)
72    variable lsb: natural range 0 to 2;
73
74  begin
75    if reset='1' then
76      lsb := 0; -- can also be 1 or 2
77
78    elsif rising_edge(mclk) then
79      ...
80
81      if ccwNotCw='0' then
82+       if angle(13 downto 2)=anglemax_vec(13 downto 2) then
83+         angle(13 downto 2) <= (others => '0');
84        else
85+         angle(13 downto 2) <= angle(13 downto 2) + 1;
86        end if;
87      else
88+       if angle(13 downto 2)=0 then
89+         angle(13 downto 2) <= anglemax_vec(13 downto 2);
90        else
91+         angle(13 downto 2) <= angle(13 downto 2) - 1;
92        end if;
93      end if;
94
95      ...
96    end if;
97  end process;
```

- set_clock_groups (Radiant)

```
2- set_clock_groups -group [get_clocks aclk] -asynchronous
3- set_clock_groups -group [get_clocks mclk] -asynchronous
4- set_clock_groups -group [get_clocks clk_byte] -asynchronous
5- set_clock_groups -group [get_clocks pixclk] -asynchronous
```

```
2+ set_clock_groups -group [get_clocks aclk] \
3+         -group [get_clocks mclk] \
4+         -group [get_clocks clk_byte] \
5+         -group [get_clocks pixclk] \
6+         -asynchronous
```

- Synplify as most efficient unused-block eliminator

CPU Host

PHY‘s & Linux | Daemon

- AXIlite/64 (ag, ay, dc, zu), AXIlite/32 (ti), UART-over-USB (cl, nx)
- WhizniumDBE bytecode protocol
- Universal 32/64 bit character device driver for AXIlite dbeaxilite w/ min. overhead ioctl()‘s; read() / write() for serial I/O
- Standardization through Linux, Yocto/Buildroot and adaptation by vendors are excellent

runCmd general case (parameters inv/ret: K/L)

	byte 0	byte 1	...						
→	Buffer::INV	tixVController	tixVCommand	invparlen 31	...	...	... 0	CRC 15	... 0
←	ACK/NAK								
→	invpar byte 0	...	... N-1	CRC 15	... 0				
←	ACK/NAK								
→	Buffer::RET	0x00	Bufop::POLL	0x00	0x00	0x00	0x04	CRC 15	... 0
←	ACK/NAK								
←	tixVController	tixVCommand	retparlen 15	... 0	CRC 15	... 0			
→	ACK/NAK								
←	retpar byte 0	...	... N-1	CRC 15	... 0				
→	ACK/NAK								

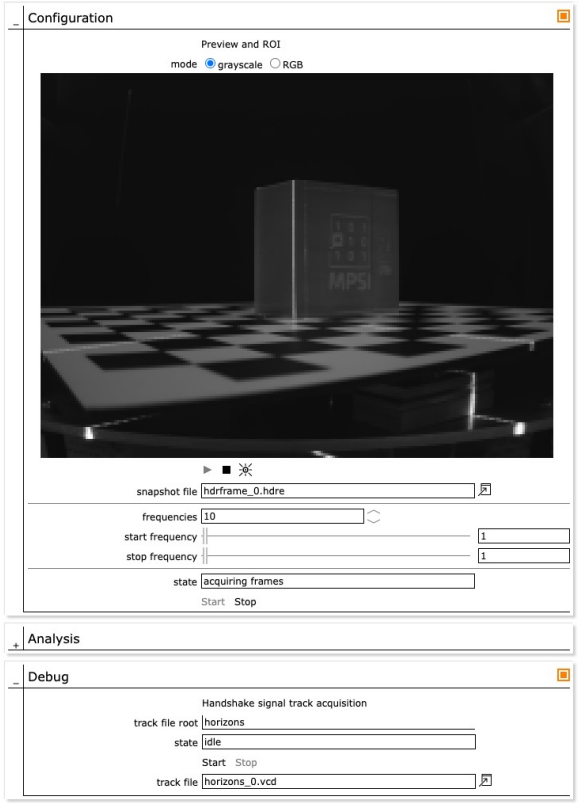
Legend

→ host to FPGA		← FPGA to host	
handshake	addressing	length	payload

CPU Host

PHY‘s & Linux | Daemon

- WhizniumSBE-managed main executable with built-in HAL functionality
- Underlying hardware (”target“) simple flag



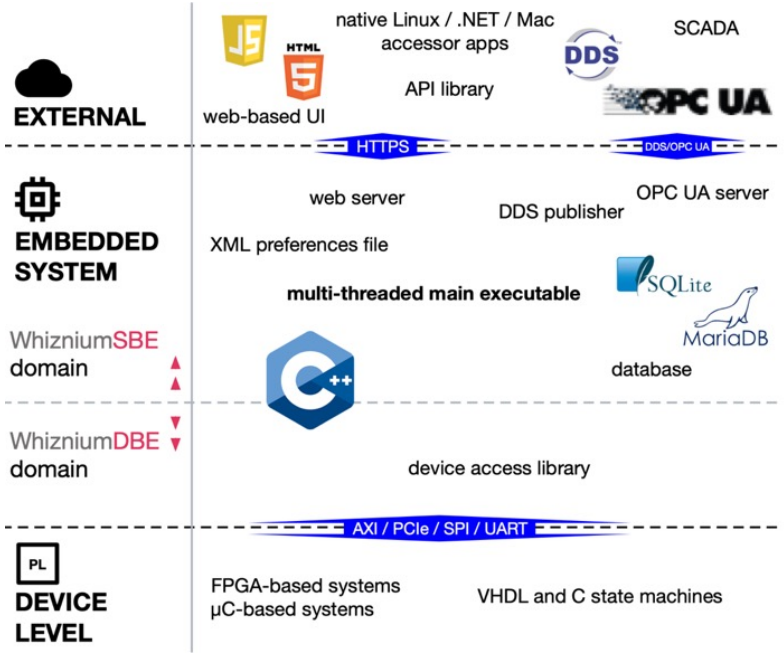

```
root@buildroot:~/whiznium/bin/wzskcmbd# ./Wzskcmbd

Welcome to Whiznium StarterKit v1.2.16!
JobWzskSrcTivsp device version 1.2.15 identified
      starting 4 job processor threads ... {133, 135, 139, 141} success
      starting 1 operation processor threads ... {146} success
      starting application server ... success
Initialization complete.

Wzskcmbd >> showSges
      + RootWzsk (1)
            + JobWzskAcqHdr/SRV (4)
                  1: idle
                  - JobWzskSrcTivsp/CLI (5)
            + JobWzskAcqPreview/SRV (6) II.2
                  2: acquiring frames
                  - JobWzskSrcTivsp/CLI (7) II.3'
            + JobWzskAcqTrackTivsp/SRV (12) III.2
                  1: idle
                  - JobWzskSrcTivsp/CLI (13) III.3'
            + JobWzskActLaser/SRV (16)
                  - JobWzskSrcTivsp/CLI (17)
            + JobWzskActRotary/SRV (18) I.2
                  2: moving
                  - JobWzskSrcTivsp/CLI (19) I.3'
            - JobWzskSrcDcvsp/SRV (27)
                  1: uninitialized
            - JobWzskSrcTivsp/SRV (31) I.3 II.3 III.3
                  2: ready
            - JobWzskSrcZuvsp/SRV (32)
                  1: uninitialized
            + SessWzsk (33)
                  + CrdWzskLlv (38, dcol)
                        1: idle
                        - PnlWzskLlvHeadbar (39)
                        + PnlWzskLlvIdent (40)
                              - JobWzskSrcTivsp/CLI (41)
                        + PnlWzskLlvRotary (48) I.1
                              - JobWzskActRotary/CLI (49) I.2'
                        + PnlWzskLlvTrackTivsp (57) III.1
                              - JobWzskAcqTrackTivsp/CLI (58) III.2'
                  + CrdWzskHwc (61, dcol)
                        1: idle
                        + PnlWzskHwcConfig (63) II.1
                              1: idle
                              - JobWzskAcqHdr/CLI (64)
                              - JobWzskAcqPreview/CLI (65) II.2'
                              - JobWzskPrcWavelet/CLI (66)
```

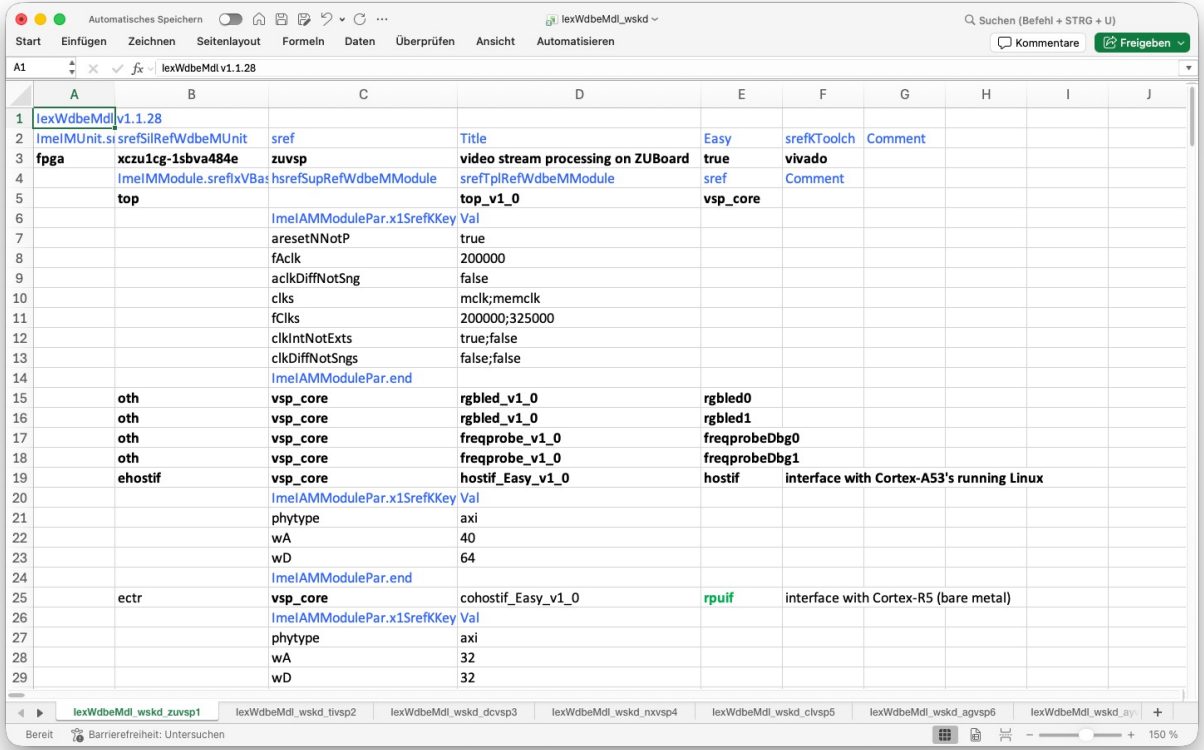
The Role of Whiznium

- Clean, ergonomic, source code structure (“tasteful naming conventions”, etc.)
- Parametrized templates for standard components (e.g. SPI, GPIO, CRC, Git-Ident; 35 and counting)
- Custom templates can interact with the model / module surroundings while a design is composed (not just simple files with placeholders)
- The applicable vendor(‘s primitives) can be an auto-derived template parameter
- Scope extends beyond the FPGA world with **WhizniumSBE** (Service Builder’s Edition)



The Role of Whiznium

- Modular structure (top-level) model file, seven variants



Automatisches Speichern — lexWdbeMdl_wskd — Suchen (Befehl + STRG + U)

Start Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen Ansicht Automatisieren — Kommentare — Freigeben

A1 | lexWdbeMdl v1.1.28

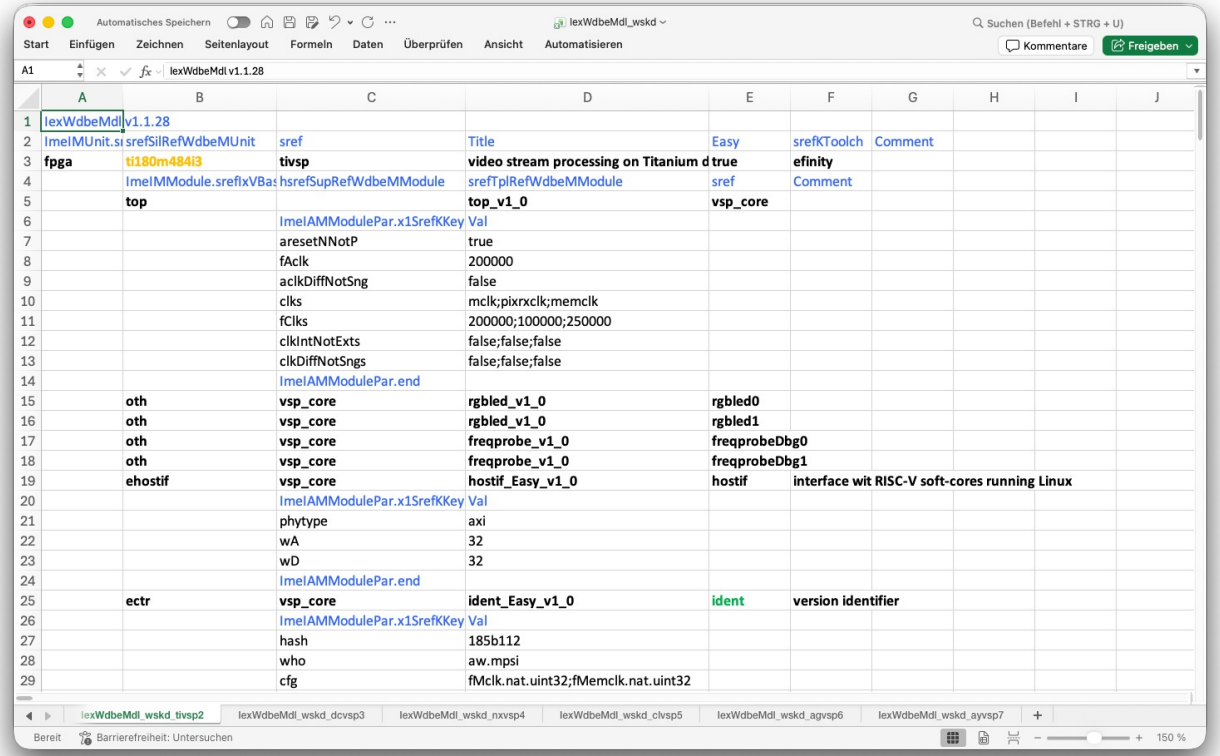
	A	B	C	D	E	F	G	H	I	J
1	lexWdbeMdl	v1.1.28								
2	ImeIMUnit.si	srefSilRefWdbeMUnit	sref	Title	Easy	srefKToolch	Comment			
3	**fpga**	**xczu1cg-1sbva484e**	**zuvsp**	**video stream processing on ZUBoard**	**true**	**vivado**				
4		ImeIMModule.srefIxVBas	hsrefSupRefWdbeMModule	srefTplRefWdbeMModule	sref	Comment				
5		**top**		**top_v1_0**	**vsp_core**					
6			ImeIAMModulePar.x1SrefKKey	Val						
7			aresetNNotP	true						
8			fAclk	200000						
9			aclkDiffNotSng	false						
10			clks	mclk;memclk						
11			fClks	200000;325000						
12			clkIntNotExts	true;false						
13			clkDiffNotSngs	false;false						
14			ImeIAMModulePar.end							
15		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled0**					
16		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled1**					
17		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg0**					
18		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg1**					
19		**ehostif**	**vsp_core**	**hostif_Easy_v1_0**	**hostif**	**interface with Cortex-A53's running Linux**				
20			ImeIAMModulePar.x1SrefKKey	Val						
21			phytype	axi						
22			wA	40						
23			wD	64						
24			ImeIAMModulePar.end							
25		ectr	**vsp_core**	cohostif_Easy_v1_0	**rpuif**	interface with Cortex-R5 (bare metal)				
26			ImeIAMModulePar.x1SrefKKey	Val						
27			phytype	axi						
28			wA	32						
29			wD	32						

lexWdbeMdl_wskd_zuvsp1 | lexWdbeMdl_wskd_tivsp2 | lexWdbeMdl_wskd_dcvsp3 | lexWdbeMdl_wskd_nxvsp4 | lexWdbeMdl_wskd_clvsp5 | lexWdbeMdl_wskd_agvsp6 | lexWdbeMdl_wskd_ay

Bereit — Barrierefreiheit: Untersuchen — 150 %

The Role of Whiznium

- Modular structure (top-level) model file, seven variants



Automatisches Speichern — lexWdbeMdl_wskd — Suchen (Befehl + STRG + U)

Start Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen Ansicht Automatisieren — Kommentare — Freigeben

A1 | lexWdbeMdl v1.1.28

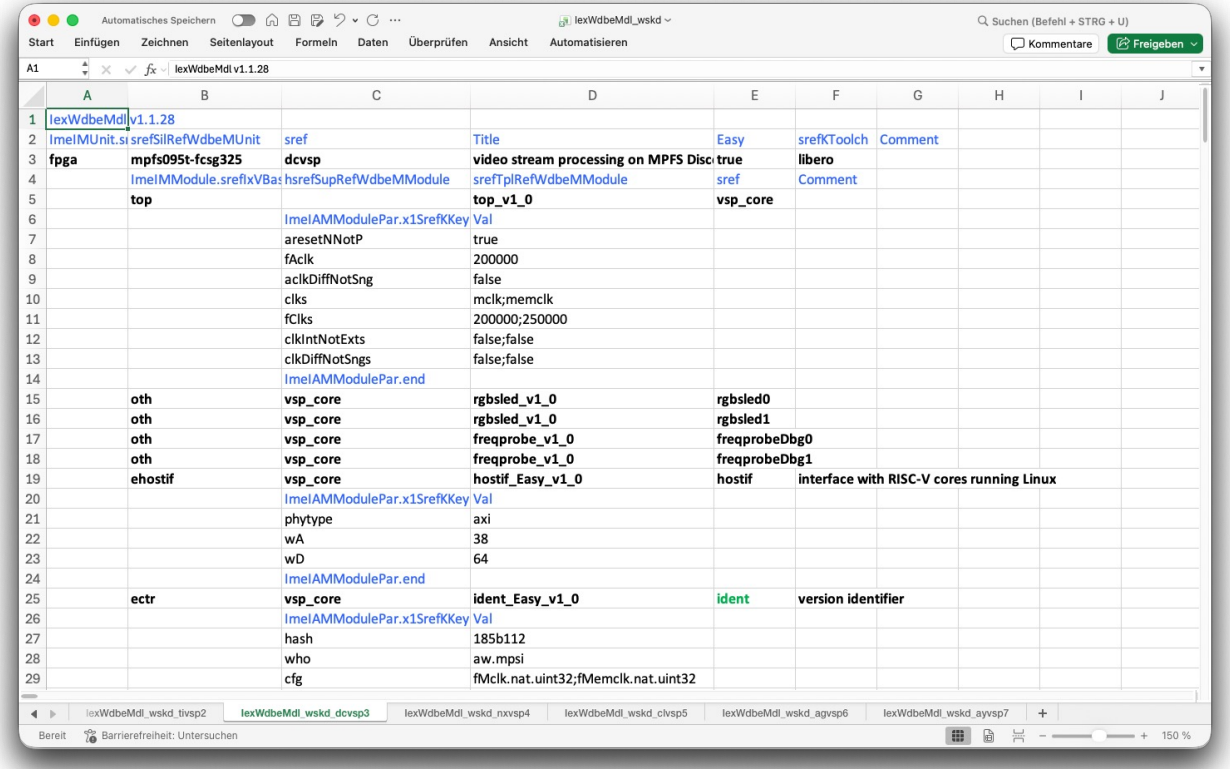
	A	B	C	D	E	F	G	H	I	J
1	lexWdbeMdl	v1.1.28								
2	ImeIMUnit.si	srefSilRefWdbeMUnit	sref	Title	Easy	srefKToolch	Comment			
3	fpga	ti180m484i3	tivsp	video stream processing on Titanium d	true	efinity				
4		ImeIMModule.srefIxVBas	hsrefSupRefWdbeMModule	srefTplRefWdbeMModule	sref	Comment				
5		top		top_v1_0	vsp_core					
6			ImeIAMModulePar.x1SrefKKey	Val						
7			aresetNNotP	true						
8			fAclk	200000						
9			aclkDiffNotSng	false						
10			clks	mclk;pixrxclk;memclk						
11			fClks	200000;100000;250000						
12			clkIntNotExts	false;false;false						
13			clkDiffNotSngs	false;false;false						
14			ImeIAMModulePar.end							
15		oth	vsp_core	rgbled_v1_0	rgbled0					
16		oth	vsp_core	rgbled_v1_0	rgbled1					
17		oth	vsp_core	freqprobe_v1_0	freqprobeDbg0					
18		oth	vsp_core	freqprobe_v1_0	freqprobeDbg1					
19		ehostif	vsp_core	hostif_Easy_v1_0	hostif	interface wit RISC-V soft-cores running Linux				
20			ImeIAMModulePar.x1SrefKKey	Val						
21			phytype	axi						
22			wA	32						
23			wD	32						
24			ImeIAMModulePar.end							
25		ectr	vsp_core	ident_Easy_v1_0	ident	version identifier				
26			ImeIAMModulePar.x1SrefKKey	Val						
27			hash	185b112						
28			who	aw.mpsi						
29			cfg	fMclk.nat.uint32;fMemclk.nat.uint32						

lexWdbeMdl_wskd_tivsp2 | lexWdbeMdl_wskd_dcvsp3 | lexWdbeMdl_wskd_nxvsp4 | lexWdbeMdl_wskd_clvsp5 | lexWdbeMdl_wskd_agvsp6 | lexWdbeMdl_wskd_ayvsp7

Bereit — Barrierefreiheit: Untersuchen — 150 %

The Role of Whiznium

- Modular structure (top-level) model file, seven variants



Automatisches Speichern — lexWdbeMdl_wskd — Suchen (Befehl + STRG + U)

Start Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen Ansicht Automatisieren — Kommentare — Freigeben

A1 | lexWdbeMdl v1.1.28

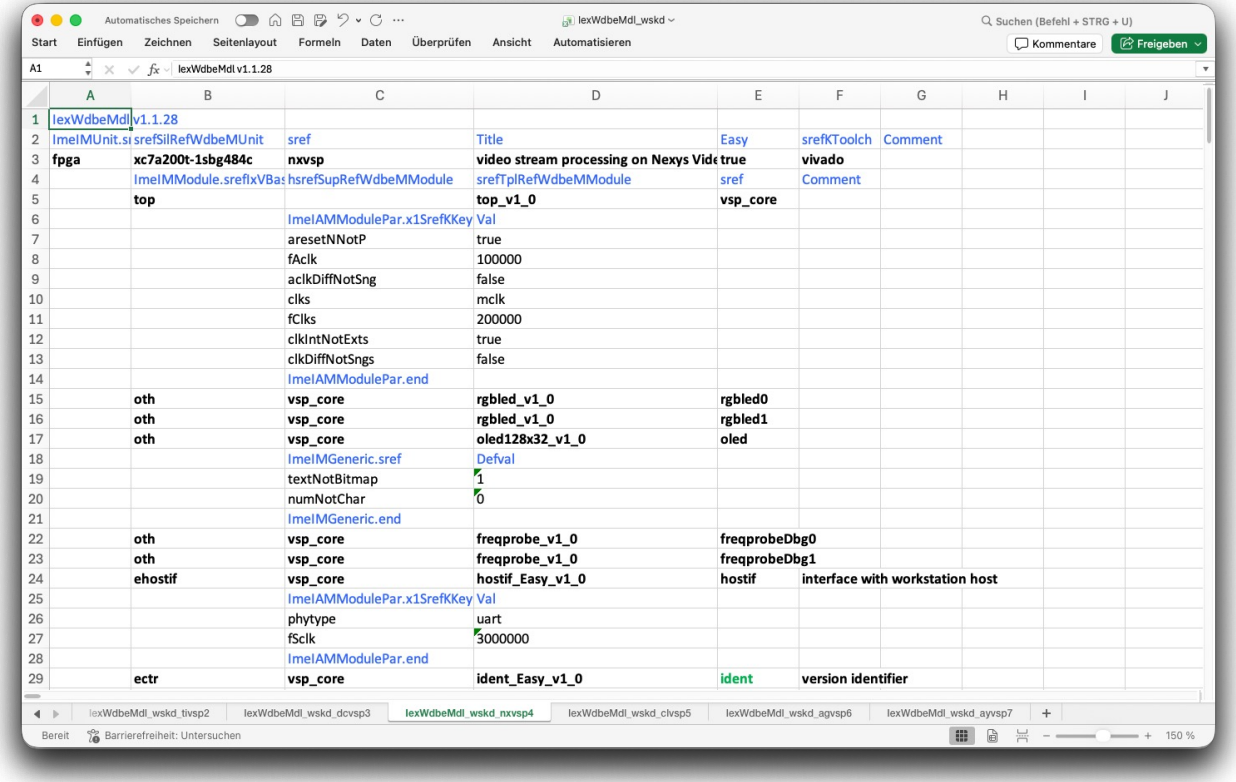
	A	B	C	D	E	F	G	H	I	J
1	lexWdbeMdl	v1.1.28								
2	ImelMUnit.si	srefSilRefWdbeMUnit	sref	Title	Easy	srefKToolch	Comment			
3	**fpga**	**mpfs095t-fcsg325**	**dcvsp**	**video stream processing on MPFS Disc**	**true**	**libero**				
4		ImelMModule.sreflxVBas	hsrefSupRefWdbeMModule	srefTplRefWdbeMModule	sref	Comment				
5		**top**		**top_v1_0**	**vsp_core**					
6			ImelAMModulePar.x1SrefKKey	Val						
7			aresetNNotP	true						
8			fAclk	200000						
9			aclkDiffNotSng	false						
10			clks	mclk;memclk						
11			fClks	200000;250000						
12			clkIntNotExts	false;false						
13			clkDiffNotSngs	false;false						
14			ImelAMModulePar.end							
15		**oth**	**vsp_core**	**rgbsled_v1_0**	**rgbsled0**					
16		**oth**	**vsp_core**	**rgbsled_v1_0**	**rgbsled1**					
17		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg0**					
18		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg1**					
19		**ehostif**	**vsp_core**	**hostif_Easy_v1_0**	**hostif**	**interface with RISC-V cores running Linux**				
20			ImelAMModulePar.x1SrefKKey	Val						
21			phytype	axi						
22			wA	38						
23			wD	64						
24			ImelAMModulePar.end							
25		**ectr**	**vsp_core**	**ident_Easy_v1_0**	**ident**	**version identifier**				
26			ImelAMModulePar.x1SrefKKey	Val						
27			hash	185b112						
28			who	aw.mpsi						
29			cfg	fMclk.nat.uint32;fMemclk.nat.uint32						

lexWdbeMdl_wskd_tivsp2 | lexWdbeMdl_wskd_dcvsp3 | lexWdbeMdl_wskd_nxvsp4 | lexWdbeMdl_wskd_clvsp5 | lexWdbeMdl_wskd_agvsp6 | lexWdbeMdl_wskd_ayvsp7

Bereit — Barrierefreiheit: Untersuchen — 150 %

The Role of Whiznium

- Modular structure (top-level) model file, seven variants



Automatisches Speichern — lexWdbeMdl_wskd — Suchen (Befehl + STRG + U)

Start | Einfügen | Zeichnen | Seitenlayout | Formeln | Daten | Überprüfen | Ansicht | Automatisieren — Kommentare — Freigeben

A1: lexWdbeMdl v1.1.28

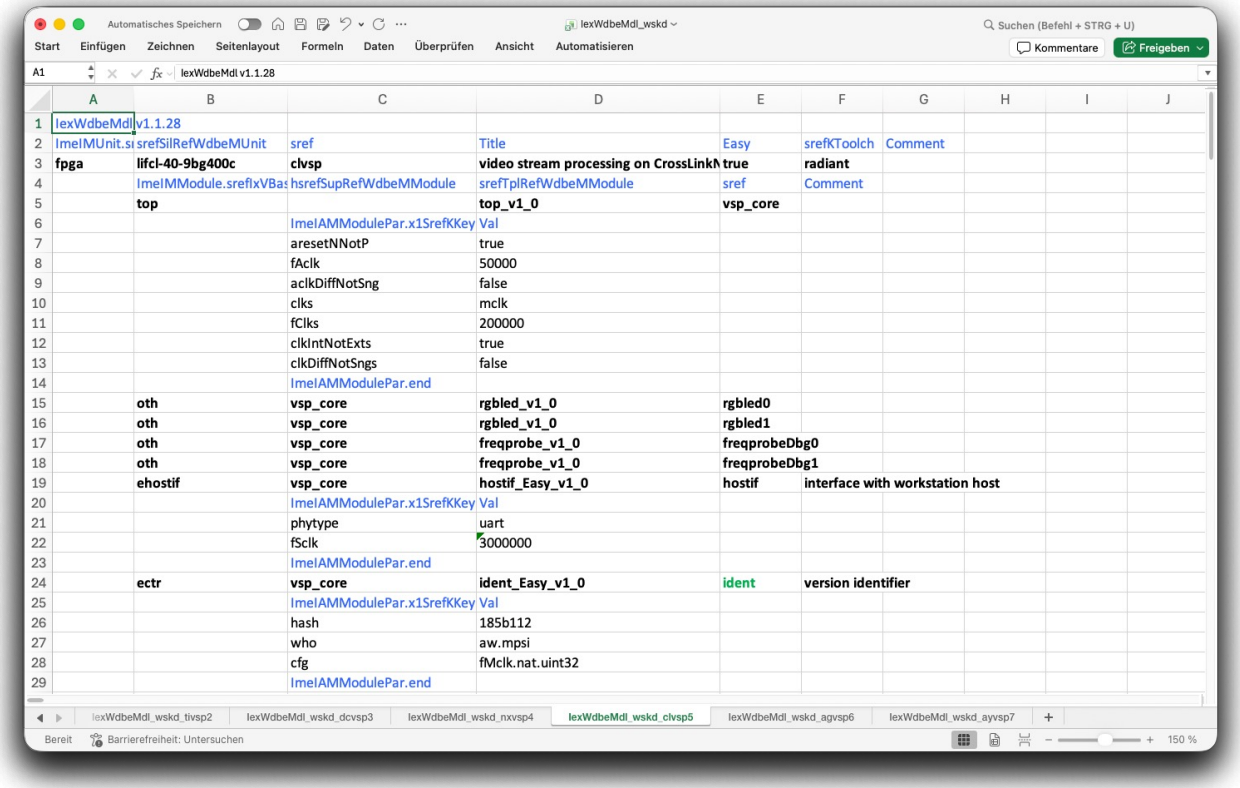
	A	B	C	D	E	F	G
1	lexWdbeMdl	v1.1.28					
2	ImeIMUnit.si	srefSilRefWdbeMUnit	sref	Title	Easy	srefKToolch	Comment
3	**fpga**	**xc7a200t-1sbg484c**	**nxvsp**	**video stream processing on Nexys Vide**	**true**	**vivado**	
4		ImeIMModule.srefIxVBas	hsrefSupRefWdbeMModule	srefTplRefWdbeMModule	sref	Comment	
5		**top**		**top_v1_0**	**vsp_core**		
6			ImeIAMModulePar.x1SrefKKey	Val			
7			aresetNNotP	true			
8			fAclk	100000			
9			aclkDiffNotSng	false			
10			clks	mclk			
11			fClks	200000			
12			clkIntNotExts	true			
13			clkDiffNotSngs	false			
14			ImeIAMModulePar.end				
15		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled0**		
16		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled1**		
17		**oth**	**vsp_core**	**oled128x32_v1_0**	**oled**		
18			ImeIMGeneric.sref	Defval			
19			textNotBitmap	1			
20			numNotChar	0			
21			ImeIMGeneric.end				
22		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg0**		
23		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg1**		
24		**ehostif**	**vsp_core**	**hostif_Easy_v1_0**	**hostif**	**interface with workstation host**	
25			ImeIAMModulePar.x1SrefKKey	Val			
26			phytype	uart			
27			fSclk	3000000			
28			ImeIAMModulePar.end				
29		**ectr**	**vsp_core**	**ident_Easy_v1_0**	**ident**	**version identifier**	

lexWdbeMdl_wskd_tivsp2 | lexWdbeMdl_wskd_dcvsp3 | lexWdbeMdl_wskd_nxvsp4 | lexWdbeMdl_wskd_clvsp5 | lexWdbeMdl_wskd_agvsp6 | lexWdbeMdl_wskd_ayvsp7

Bereit — Barrierefreiheit: Untersuchen — 150 %

The Role of Whiznium

- Modular structure (top-level) model file, seven variants



Automatisches Speichern — lexWdbeMdl_wskd — Suchen (Befehl + STRG + U)

Start | Einfügen | Zeichnen | Seitenlayout | Formeln | Daten | Überprüfen | Ansicht | Automatisieren — Kommentare — Freigeben

A1: lexWdbeMdl v1.1.28

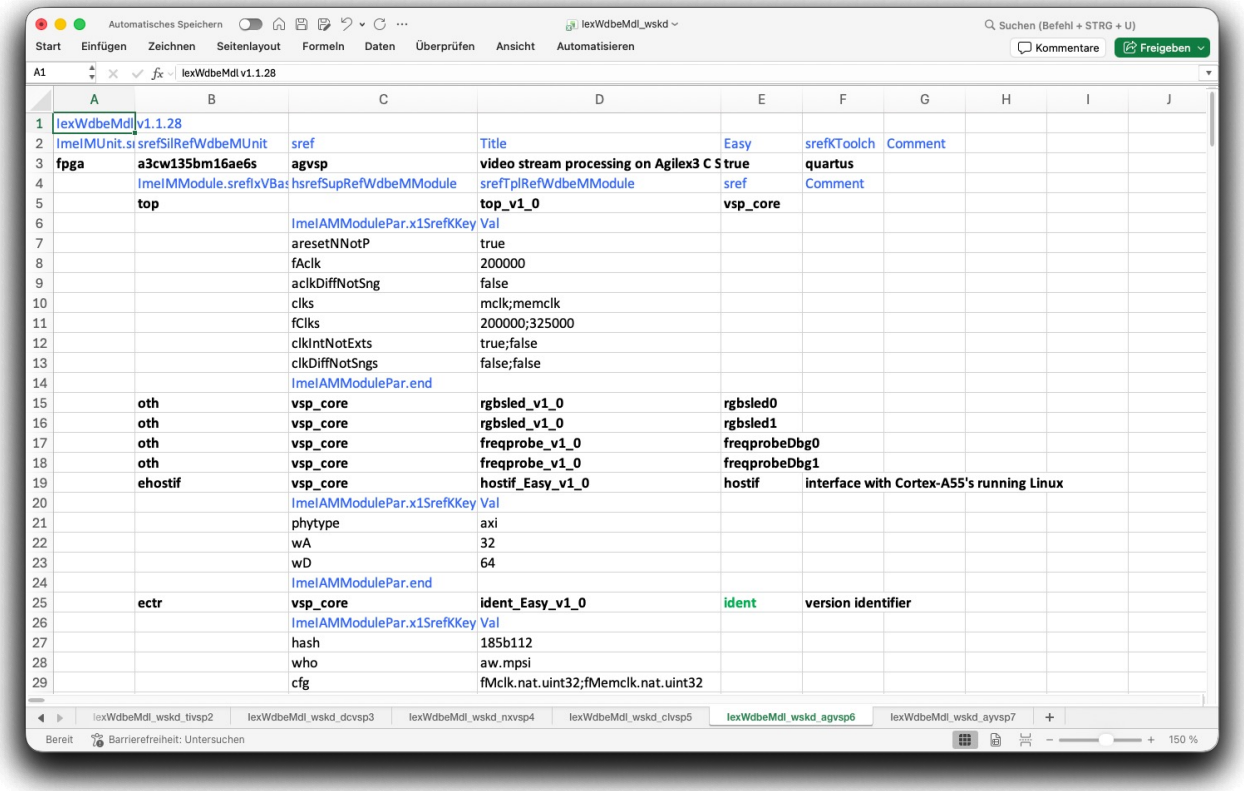
	A	B	C	D	E	F	G
1	lexWdbeMdl	v1.1.28					
2	ImeIMUnit.si	srefSilRefWdbeMUnit	sref	Title	Easy	srefKToolch	Comment
3	**fpga**	**lifcl-40-9bg400c**	**clvsp**	**video stream processing on CrossLinkN**	**true**	**radiant**	
4		ImeIMModule.srefIxVBas	hsrefSupRefWdbeMModule	srefTplRefWdbeMModule	sref	Comment	
5		**top**		**top_v1_0**	**vsp_core**		
6			ImeIAMModulePar.x1SrefKKey	Val			
7			aresetNNotP	true			
8			fAclk	50000			
9			aclkDiffNotSng	false			
10			clks	mclk			
11			fClks	200000			
12			clkIntNotExts	true			
13			clkDiffNotSngs	false			
14			ImeIAMModulePar.end				
15		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled0**		
16		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled1**		
17		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg0**		
18		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg1**		
19		**ehostif**	**vsp_core**	**hostif_Easy_v1_0**	**hostif**	**interface with workstation host**	
20			ImeIAMModulePar.x1SrefKKey	Val			
21			phytype	uart			
22			fSclk	3000000			
23			ImeIAMModulePar.end				
24		**ectr**	**vsp_core**	**ident_Easy_v1_0**	**ident**	**version identifier**	
25			ImeIAMModulePar.x1SrefKKey	Val			
26			hash	185b112			
27			who	aw.mpsi			
28			cfg	fMclk.nat.uint32			
29			ImeIAMModulePar.end				

lexWdbeMdl_wskd_tivsp2 | lexWdbeMdl_wskd_dcvsp3 | lexWdbeMdl_wskd_nxvsp4 | lexWdbeMdl_wskd_clvsp5 | lexWdbeMdl_wskd_agvsp6 | lexWdbeMdl_wskd_ayvsp7

Bereit — Barrierefreiheit: Untersuchen — 150 %

The Role of Whiznium

- Modular structure (top-level) model file, seven variants



Automatisches Speichern — lexWdbeMdl_wskd — Suchen (Befehl + STRG + U)

Start Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen Ansicht Automatisieren — Kommentare — Freigeben

A1 | lexWdbeMdl v1.1.28

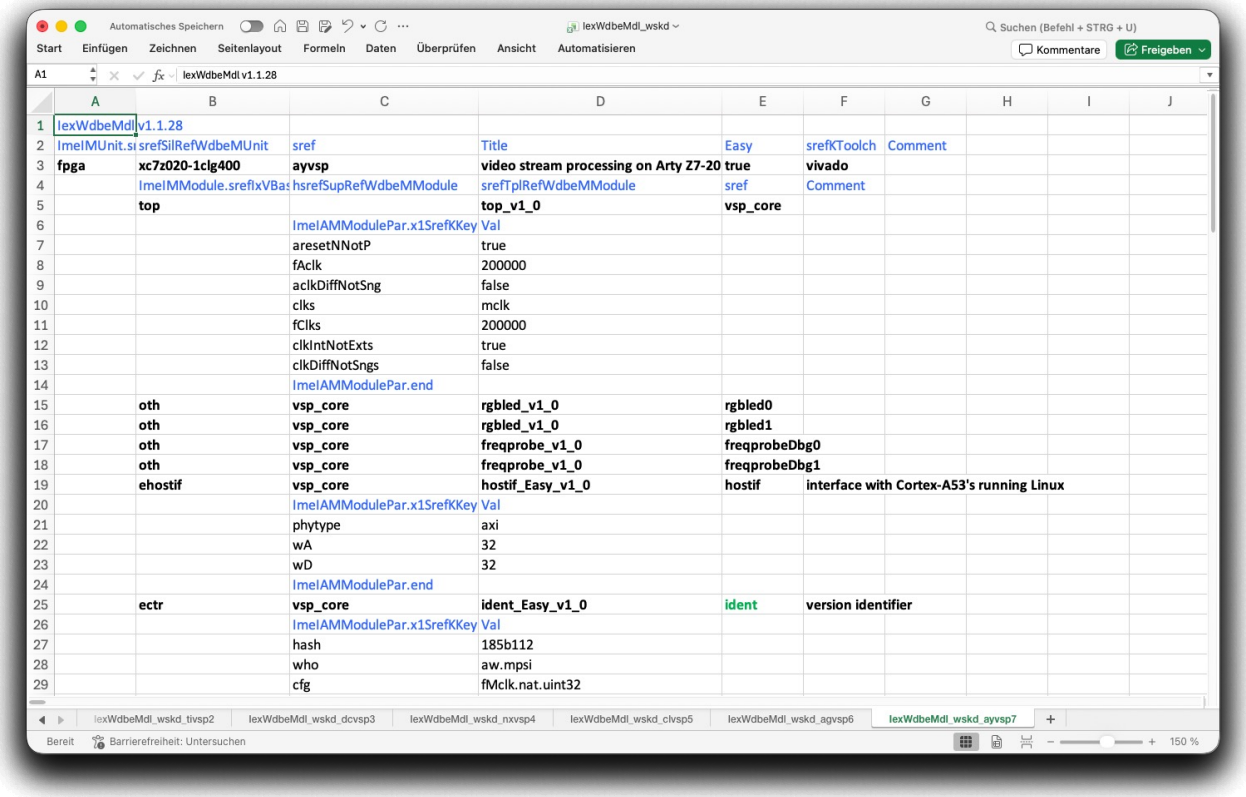
	A	B	C	D	E	F	G	H	I	J
1	lexWdbeMdl	v1.1.28								
2	ImeIMUnit.si	srefSilRefWdbeMUnit	sref	Title	Easy	srefKToolch	Comment			
3	**fpga**	**a3cw135bm16ae6s**	**agvsp**	**video stream processing on Agilex3 C S**	**true**	**quartus**				
4		ImeIMModule.srefIxVBas	hsrefSupRefWdbeMModule	srefTplRefWdbeMModule	sref	Comment				
5		**top**		**top_v1_0**	**vsp_core**					
6			ImeIAMModulePar.x1SrefKKey	Val						
7			aresetNNotP	true						
8			fAclk	200000						
9			aclkDiffNotSng	false						
10			clks	mclk;memclk						
11			fClks	200000;325000						
12			clkIntNotExts	true;false						
13			clkDiffNotSngs	false;false						
14			ImeIAMModulePar.end							
15		**oth**	**vsp_core**	**rgbsled_v1_0**	**rgbsled0**					
16		**oth**	**vsp_core**	**rgbsled_v1_0**	**rgbsled1**					
17		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg0**					
18		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg1**					
19		**ehostif**	**vsp_core**	**hostif_Easy_v1_0**	**hostif**	**interface with Cortex-A55's running Linux**				
20			ImeIAMModulePar.x1SrefKKey	Val						
21			phytype	axi						
22			wA	32						
23			wD	64						
24			ImeIAMModulePar.end							
25		**ectr**	**vsp_core**	**ident_Easy_v1_0**	**ident**	**version identifier**				
26			ImeIAMModulePar.x1SrefKKey	Val						
27			hash	185b112						
28			who	aw.mpsi						
29			cfg	fMclk.nat.uint32;fMemclk.nat.uint32						

lexWdbeMdl_wskd_tivsp2 | lexWdbeMdl_wskd_dcvsp3 | lexWdbeMdl_wskd_nxvsp4 | lexWdbeMdl_wskd_clvsp5 | lexWdbeMdl_wskd_agvsp6 | lexWdbeMdl_wskd_ayvsp7

Bereit — Barrierefreiheit: Untersuchen — 150 %

The Role of Whiznium

- Modular structure (top-level) model file, seven variants



Automatisches Speichern — lexWdbeMdl_wskd — Suchen (Befehl + STRG + U)

Start Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen Ansicht Automatisieren — Kommentare — Freigeben

A1 | lexWdbeMdl v1.1.28

	A	B	C	D	E	F	G	H	I	J
1	lexWdbeMdl	v1.1.28								
2	ImelMUnit.si	srefSilRefWdbeMUnit	sref	Title	Easy	srefKToolch	Comment			
3	**fpga**	**xc7z020-1clg400**	**ayvsp**	**video stream processing on Arty Z7-20**	**true**	**vivado**				
4		ImelMModule.srefIxVBas	hsrefSupRefWdbeMModule	srefTplRefWdbeMModule	sref	Comment				
5		**top**		**top_v1_0**	**vsp_core**					
6			ImelAMModulePar.x1SrefKKey	Val						
7			aresetNNotP	true						
8			fAclk	200000						
9			aclkDiffNotSng	false						
10			clks	mclk						
11			fClks	200000						
12			clkIntNotExts	true						
13			clkDiffNotSngs	false						
14			ImelAMModulePar.end							
15		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled0**					
16		**oth**	**vsp_core**	**rgbled_v1_0**	**rgbled1**					
17		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg0**					
18		**oth**	**vsp_core**	**freqprobe_v1_0**	**freqprobeDbg1**					
19		**ehostif**	**vsp_core**	**hostif_Easy_v1_0**	**hostif**	**interface with Cortex-A53's running Linux**				
20			ImelAMModulePar.x1SrefKKey	Val						
21			phytype	axi						
22			wA	32						
23			wD	32						
24			ImelAMModulePar.end							
25		**ectr**	**vsp_core**	**ident_Easy_v1_0**	**ident**	**version identifier**				
26			ImelAMModulePar.x1SrefKKey	Val						
27			hash	185b112						
28			who	aw.mpsi						
29			cfg	fMclk.nat.uint32						

lexWdbeMdl_wskd_tivsp2 | lexWdbeMdl_wskd_dcvsp3 | lexWdbeMdl_wskd_nxvsp4 | lexWdbeMdl_wskd_clvsp5 | lexWdbeMdl_wskd_agvsp6 | lexWdbeMdl_wskd_ayvsp7

Bereit — Barrierefreiheit: Untersuchen — 150 %

- Parametrized WhizniumDBE module templates used in the project

axislave_v1_0	AXI lite slave with AXI stream interface
cohostif_Easy_v1_0	universal PHY co-host interface (e.g. RPU+CPU)
crcspec_v3_0	CRC calculator with AXI stream interface
ddrmux_Easy_v1_0	DDR memory access multiplexer (cross-vendor)
dpbram_v1_0	dual-port block RAM (cross-vendor)
freqprobe_v1_0	word to hex / BCD frequency converter
fsmtrack_Easy_v1_0	FSM state tracker (cross-vendor)
gptrack_Easy_v1_0	general purpose signal tracker (cross-vendor)
hostif_Easy_v1_0	universal PHY host interface
i2cmaster_v1_0	I2C transceiver
ident_Easy_v1_0	version idenfitication (Git hash and generics)
mult_v2_0	signed/unsigned multiplier with configurable latency

- (cont'd)

oled128x32_v1_0	bitmap/char driver for SSD1306 OLED display
rgbled_v1_0	driver for RGB LED (PWM)
rgbsled_v1_0	driver for WS2812B RGB LED
spbram_v1_0	single-port block RAM (cross-vendor)
spimaster_v1_0	SPI master with AXI stream interface
sprsaxisfwd_v1_0	unstable ingress sparse AXI stream forwarder with no back-pressure allowed
timeout_v1_0	time-out counter
tkclksrc_Easy_v1_0	10kHz clock source
top_v1_0	top-level module (cross-vendor)
uartrx_v2_0	UART receiver with AXI stream interface
uarttx_v2_0	UART transmitter with AXI stream interface

The Role of Whiznium

- Better VHDL: not enough initialization (MPFS antifuse-based default ‘1’), too much initialization (stateXyz := stateXyzInit in declaration overrides process :=)
- Added constraints file formats
- Created vendor-abstracting wrappers
- Improved dbeaxilite module as RISC-V is more sensitive to mistakes kernel vs. user space than aarch64

Conclusion

- Whiznium CV demonstrator great reality check for vendor agnosticity

- How to keep your design flexible and platform-agnostic
  - parametrize
  - simulate components outside of vendor tools
  - prepare for block memory variety
  - **use Whiznium**
  - use GUI as little as possible
  - use vendor IP core generators

Conclusion

- How to design multi-platform
  - keep Whiznium model description and code base side-by-side (no separate Git branches)

- How to approach a new platform
  - talk to someone using it for first step
  - use the vendor example designs as base (how to do timing, clock architecture, wire up things)

Outlook

- More platforms
- LLM-assisted workflow to implement requirement updates on Whiznium model level … leaving the deterministic code generation by Whiznium in place
- Approach
  - Low-rank adaptation (LoRA) of Qwen2.5-Coder for Whiznium model files
  - Cloud training based on curated model <-> code pairs
  - Ryzen 7 (48 GB DDR4 SDRAM) + GTX 3090 (24 GB GDDR6X SDRAM) local workstation

Resources

- Computer Vision project sources
  https://github.com/mpsitech/wzsk-Whiznium-StarterKit
  https://github.com/mpsitech/wskd-Whiznium-StarterKit-Device

- Whiznium installation instructions
  https://github.com/mpsitech/The-Whiznium-Documentation

- Whiznium knowledge base (technical deep dives on specific subjects including on each 3D laser scanner variant)
  https://mpsitech.github.io/Whiznium-Knowledge-Base

- WhizniumSBE and WhizniumDBE references (info e.g. on model file structure and module/IP templates)
  https://mpsitech.github.io/The-WhizniumSBE-Reference
  https://mpsitech.github.io/The-WhizniumDBE-Reference

- Some more presentations on the topic
  https://www.mpsitech.com/documentation/presentations

Thank You!

Questions?

Also, feel free to connect.

- https://www.linkedin.com/in/wirthmua
- https://github.com/mpsitech

Alexander Wirthmueller
Founder & Director

Phone: +49 (89) 4524 3826
Mobile: +49 (175) 918 5480
E-Mail: aw@mpsitechnologies.com

MPSI Technologies GmbH
Altmuehlstrasse 1
80638 Munich
Germany
www.mpsitechnologies.com

MPSI TECHNOLOGIES

- Applied Physics background
- FPGA-SoC designs (all major vendors)
- Hand-tuned DSP algorithms
- Tightly integrated CPU+FPGA solutions
- Comms, RADAR, Computer Vision & more

WhizniumSBE and WhizniumDBE

Open-Source Developer Tools
for the Model-based Design
of Embedded Software