

# Andes软件解决方案加速 RISC-V AI与IoT应用开发

Simon TC Wang  
Technical Marketing Manager  
Andes Technology  
August 18, 2021

# Agenda

- AndeSight™ IDE
- AndeSoft™ BSP
- AI Software Stack
  - DSP and Vector programming
  - AndeSoft™ NN library
  - Use cases of NN model
- Andes Ecosystem





# AndeSight™ IDE

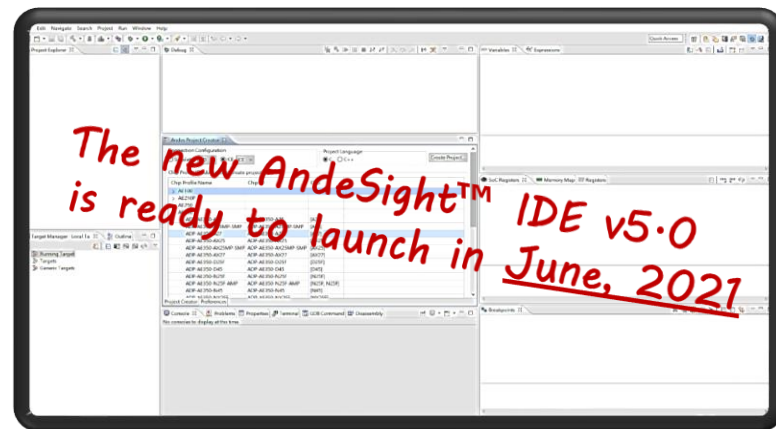
Comprehensive Development Environment

# AndeSight™ IDE for RISC-V

Accelerate your RISC-V software developments  
with the comprehensive development environment!

## AndeSight™ IDE with AndeSoft™ BSP inside

- User-friendly and easy-to-use IDE
- Highly-optimized GCC/LLVM toolchains for outstanding performance and compact memory footprint
- Abundant demos for boosting your development
- Rich RTOSes and Linux (LTP verified)
- Optimized compute library: DSP, Vector<sup>1</sup>
- Peripheral drivers for AndeShape™ platform
- Near cycle-accurate simulators: AndeSim™
- Arduino support for Andes Corvette EVB



1: features in AndeSight™ v5.0

# AndeSight™: Professional IDE

- Eclipse-based IDE, enriched from 16-year continuous development

## Bit-field display

SoC Registers Registers 33

Name	Value
cr2 (DCM_CFG)	0x00002400
cr3 (MMU_CFG)	0x60080004
VLPT	0x1 - Implemented
IVTB	0x0 - Not present
NTPPT	0x0 - 2 partitions
DE	0x0 - Little
HPTWVK	0x0 - No HPTWVK
TBLCK	0x0 - Not supported
EPSZ	0x8

cr3 (MMU\_CFG) 0x60080004

- VLPT Implemented
- IVTB Not implemented
- NTPPT Implemented

## Function profiling

gmon file: D:\AndeSight\ide\workspace\JPEG\gmon.sum  
program file: D:\AndeSight\ide\workspace\JPEG\Debug\JPEG.adx

Name (location)	Samples	%Time
Summary	995	100.0%
jdcint.c	371	37.29%
jpeg_idct_islow	371	37.29%
jdhuft.c	152	15.28%
decode_mcu	101	10.15%
jpeg_fill_bit_buffer	44	4.42%
jpeg_huff_decode	4	0.4%
jpeg_make_d_derived_tbl	3	0.3%
jdcolor.c	149	14.97%
ycc_rgb_convert	148	14.87%
build_ycc_rgb_table	1	0.1%

## Meta linker script editor

Available Items

```
DEFINE
USER_SECTIONS
LOAD_ROM 0x0000
EXEC_ROM 0x0000
Input Sections
+ISR
.section
ADDR
LOADADDR
STACK
VAR
ALIGN
Group Input Section Pattern
EXEC OVERLAY_ROM 0x0000 OVERLAY 0
```

Sections

```
USER_SECTIONS_vector
FLASH 0x00000000 0x00100000
EXEC 0x00000000
VAR_ILM_BASE = 0x00600000
VAR_DLM_BASE = 0x00700000
VAR_ILM_SIZE = 0x00010000
VAR_DLM_SIZE = 0x00010000
(*.vector)
*(+RO)
SDRAM 0x80000000 0x00800000
LOADADDR NEXT __data_lmstart
ADDR NEXT __data_start
*(+RW,+ZI)
STACK = 0x80000000
```

## RTOS awareness

Breakpoints Console EventList-13 TaskList-13

task name number priority start of stack top of stack status

task name	number	priority	start of stack	top of stack	status
TaskWav	0	2	0x644880	0x650720	Running
IDLE	2	0	0x651a20	0x655990	Ready
TaskEmp	1	2	0x650980	0x651870	Blocked
DMA BH	3	8	0x304860	0x305370	Suspended

FreeRTOS Task List

FreeRTOS Event List

queue name	handler address	max length	item size	messages waiting	waiting Tx	waiting Rx
queue	0x655e40	1	0	1	0	0
queue	0x655b00	1	0	1	0	0
queue	0x655b00	65535	0	127	0	0
queue	0x655e80	65535	0	0	0	1

Tasks Waiting Rx

task name	number
DMA BH	3

## Custom plugin

Custom plugin window showing a circuit diagram and a waveform.

## Function code size

djpeg.c

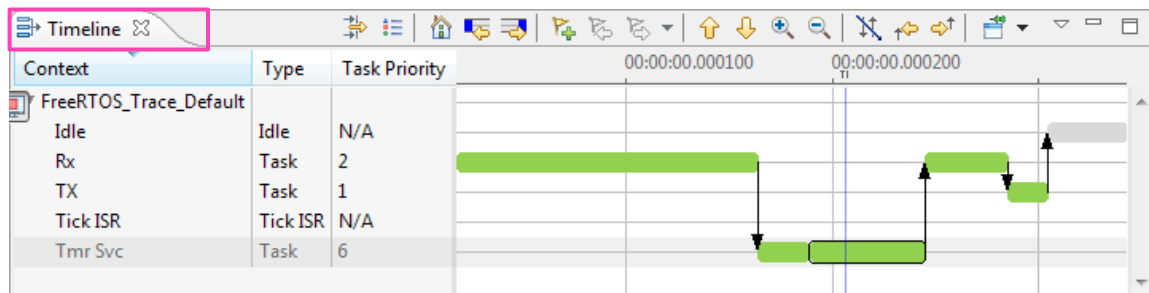
```
int djpeg_main(int argc, char **argv) //Forward reference
char* apBmp[8];
#define NDS32_ISIZE (0x07 << 6)
#define NDS32_DSIZE (0x07 << 6)
int main(void)
#include "NABLE_CACHE
unsigned int dcm_cfg, icm_cfg, cache_ctl;
dcm_cfg = nds32_mfar(NDS32_SR_DCM_CFG);
icm_cfg = nds32_mfar(NDS32_SR_ICM_CFG);
```

Binary: /FEG/Debug/JPEG.adx -> Total Function Code Size: 46928

Name	Size	File	Line	Path
ldc_off	24	ldc.c	59	d:\AndeSight\AndeSight
ldc_sel_framebase	60	ldc.c	69	d:\AndeSight\AndeSight
main	176	main.c	192	d:\AndeSight\AndeSight
make_runny_pointers	108	jdmain.c	396	d:\AndeSight\AndeSight
master_selection	464	jdmaster.c	288	d:\AndeSight\AndeSight
median_cut	252	jdquant.c	424	d:\AndeSight\AndeSight
merged_lv_upsample	48	jdmerge.c	193	d:\AndeSight\AndeSight
merged_zv_upsample	164	jdmerge.c	144	d:\AndeSight\AndeSight

# FreeRTOS Timeline Analyzer

- Visualize the runtime execution behaviors of **task**, **interrupts** and **events** within a period of time<sup>1</sup>



Timestamp	Context	Context Type	Events	Details
<srch>	<srch>	<srch>	<srch>	<srch>
00:00:00.000 164	Tmr Svc	Task	Task Switch In	Task 'Tmr Svc' context switch in and get running
00:00:00.000 189	Tmr Svc	Task	Task Switch In	Task 'Tmr Svc' context switch in and get running
00:00:00.000 202	Tmr Svc	Task	xQueueRecv	Queue='TmrQ', Status=SUCCESS
00:00:00.000 219	Tmr Svc	Task	xQueueRecv	Queue='TmrQ', Status=FAIL, TicksToWait=0
00:00:00.000 245	Rx	Task	Task Switch In	Task 'Rx' context switch in and get running

Context	Type	Run Count	Run Frequency	Last Run Time
Idle	Idle	3426	942.98 Hz	0.000991 s
Rx	Task	133	36.61 Hz	0.000984 s
TX	Task	41	11.28 Hz	0.000018 s
Tick ISR	Tick ISR	3549	976.83 Hz	0.000009 s
Tmr Svc	Task	3	.83 Hz	0.000075 s

1: EVB target only

# Multicore Development Support

- Build and debug the multicore software with separate configurations by simply creating multicore projects

The screenshot displays the AndeSight IDE interface for debugging a multicore application. The main window is titled "Debug - multicore-demo/core1/src/core1.c - AndeSight".

**Annotations:**

- Debugging actions for a single core:** A pink box highlights the "Debug" toolbar, with a pink arrow pointing to the "Breakpoint" icon.
- Context values when debugging core1:** A pink box highlights the "Expressions (x) Variables" and "Memory" panels.

**Project Structure:** The "Project Explorer" on the left shows a project named "multicore-demo" with two cores:

- core0:** Thread #1 (Suspended: Breakpoint) at main() at core0.c:17 0x100054.
- core1:** Thread #1 (Suspended: Breakpoint) at main() at core1.c:16 0x100054.

**Code Editor:** The "Code Editor" shows the source code for core1.c:

```
16 volatile int i = 0, j = 0;
17 // puts("!!!Hello World!!!");
18
19 for (i=0; i < 10; )
20 {
21     if (!*flag)
22     {
23         printf("I'm Core1 !!!, %d\n", i);
24         for (j = 0; j < 0x10000; j++); //delay loop
25         i++;
26         *flag = 1;
27     }
28 }
```

**Memory Panel:** The "Memory" panel shows the memory dump for core1 at address 0x100054:

Address	Hex	ASCII
0x00100054	C402C602 803FA7AB	.E.Ä«\$?.
0x0010005C	0007A023 00012623	# ..#8..
0x00100064	47A54732 02E7CE63	2G¥Gçİç.
0x0010006C	803FA7AB DBED439C	«\$? ..çİÜ
0x00100074	153745B2 05130010	«E7. ....
0x0010007C	28D9CC55 A029C402	ÄİÜ(.Ä)
0x00100084	078547A2 00F12423	çG. #5ñ.
0x0010008C	67C14722 FEF74AE3	"GÄgäJ+p
0x00100094	078547B2 A7ABC63E	2ç →ççç

**SoC Registers:** The "SoC Registers" panel shows a table of registers:

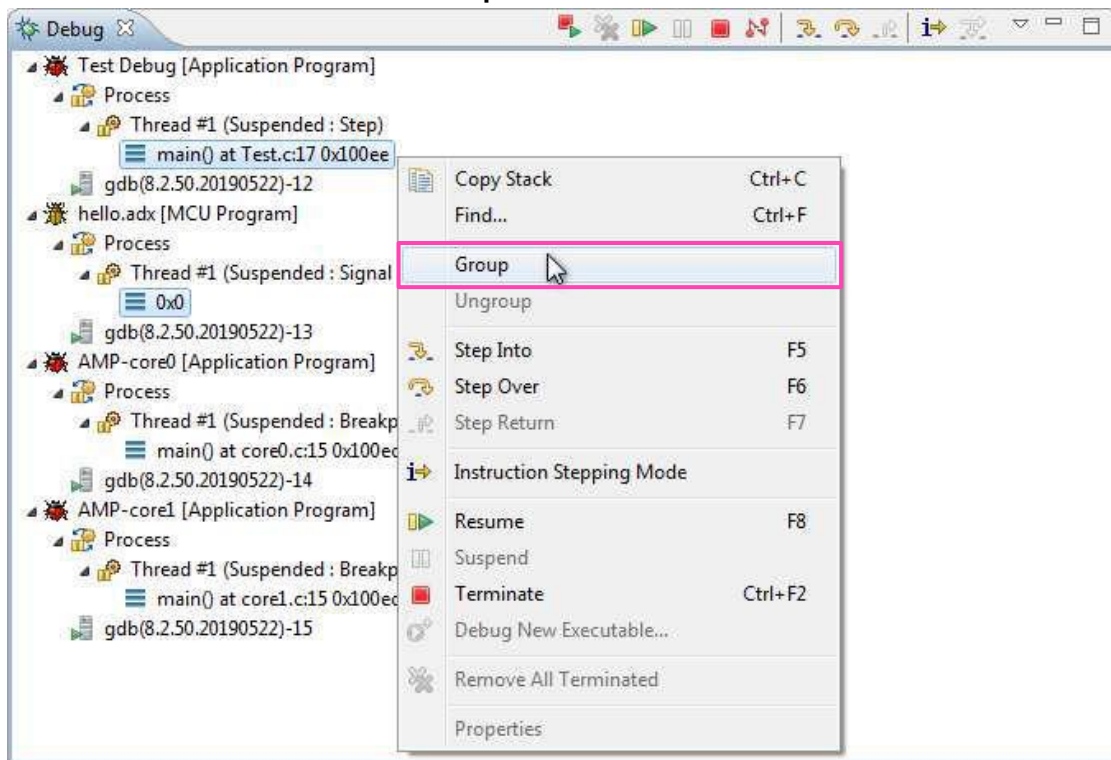
Name	Value	Address	Description
BMC			AXI Bus Inte
AHB decoder			AHB-Lite de
Ethernet MAC			Ethernet M
ISR	0x0	0x00100000	Interrupt Statu
IME	0x0	0x00100004	Interrupt Enabl
MAC_MADR	0x0	0x00100008	MAC Most Sig
MAC_LADR	0x0	0x0010000c	MAC Least Sig

**Console:** The "Console" panel shows the output of the program:

```
[28] multicore-demo-core1 [Application Program] multicore-demo-core1.adx
```

# Core Grouping

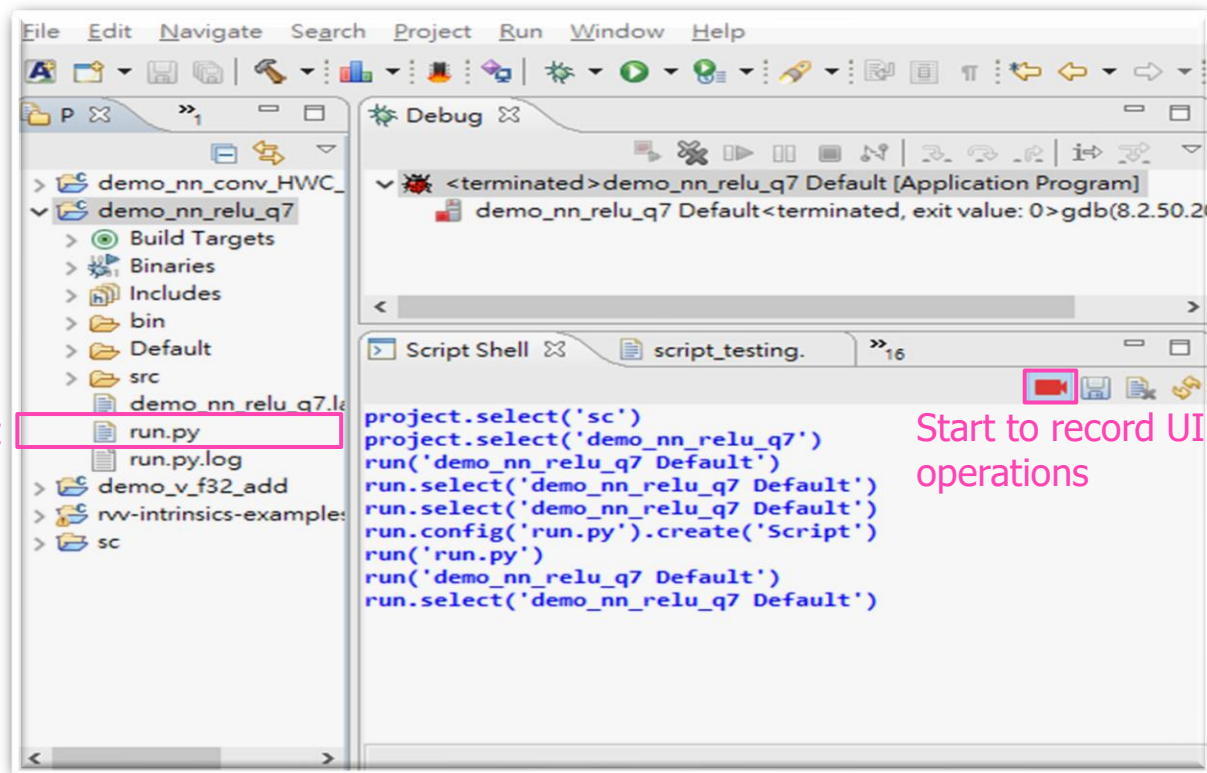
- Grouping Cores for Efficient Debugging
  - Debug commands can be sent to a specific set of cores at the same time.





# AndeSight™ Scripting

- Record and replay UI operations with Python script
  - Test automation
  - Issue reproduction
  - ...





# AndeSoft™ BSP

Application Building Blocks Inside of AndeSight™ IDE

# AndeSoft™ BSP: Application Building Blocks

## Fundamental

- Compiler/Toolchain are contributed to and supported officially by **GNU/LLVM** communities
- Optimized **MCUlib**, newlib and glibc
- Optimized low-level compute libraries for NN, DSP and vector processing: **libnn, libdsp, libvec**
- Fast and near cycle-accurate simulators: **AndeSim™, AndeSysC™, Qemu<sup>1</sup>**
- Debugging: **GDB and ICEman (speed-optimized OpenOCD)**
- Concise linker script and its tools, **Linker Scattering-and-Gathering (LdSaG)**
- **Bare-metal drivers and demo programs** to demo AndesCore™ features

## Real-Time Operating Systems

- Open source port on Andes: **FreeRTOS (RV32/64 UP), Zephyr (RV32/64 UP/SMP)**
- Commercial port on Andes<sup>2</sup>: **Azure RTOS ThreadX, LiteOS, RT-Thread, SylixOS**
- **RISC-V ready<sup>3</sup>**: VxWorks, µC/OS-[II/III], MyNewt, embOS, RTEMS, NuttX, seL4, uC3/Compact, AliOS Things, TencentOS Tiny, HarmonyOS, Nucleus



## Linux, Middleware and SW Framework

- **Linux kernel v4.17/v5.4 (LTS), UP/SMP, LTP verified, device drivers and advanced features: strace, ftrace, Perf, SMU, power throttling, suspend-to-RAM, CPU hotplug, HIGHMEM and kernel module**
- **U-Boot, U-Boot-SPL, OpenSBI and BBL**



1: available upon request

2: supported by commercial vendors and not included in Andes products. Users need to license from the commercial vendors.

3: known and available RISC-V architecture port but lack of Andes platform features. Not included in Andes products.

# AndeSoft™: Bare Metal



- Rich **startup demo** projects for Andes-specific features

Categories	Startup demo
<b>Interrupt</b>	PLIC, CLIC
<b>Memory</b>	MMU, PMP, PMA, cache, cache lock, ECC, bus matrix slave port
<b>Power Management</b>	PowerBrake, hibernate, WFI CPU standby/resume
<b>Programming</b>	DSP, printf UART redirect, C++ programming
<b>Misc</b>	StackSafe™, performance monitor, SMP

- **AMSI** (Andes MCU Software Interface) driver APIs
  - DMA, Flash, GPIO, I2C, PWM, RTC, SPI, UART and WDT

# AndeSoft™: Linux

- Linux Distribution and Build System\*

\*: available upon request

fedora<sup>f</sup>

debian

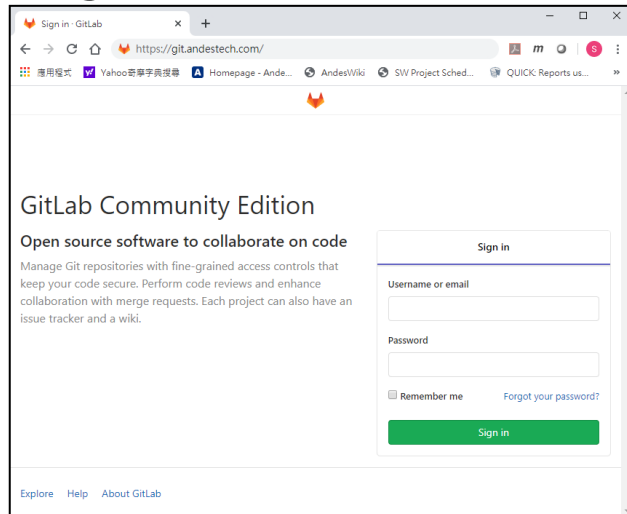
yocto  
PROJECT

OpenWrt  
Wireless Freedom

BuildRoot  
Making Embedded Linux Easy

- Andes GitLab Service for Linux Development Packages

- A public web service: [git.andestech.com](https://git.andestech.com)
- Under Andes Maintenance Program
- Linux toolchain and source code



ANDES  
TECHNOLOGY



GitLab



# AI Software Stack

DSP, Vector and NN Programming

# AI Software Stack and Ecosystem

AI applications

Signal and data processing  
pre-/post-processing for  
audio, voice, video, image, ...



AI frameworks &  
NN model format

NN model optimization

NN inference engine



AndeSoft™ LLVM compiler  
AndeSoft™ OpenCL™



AndeSoft™  
DSP Library

AndeSoft™  
Vector Library

AndeSoft™  
NN Library

AndeCore™ baseline, RVP, RVV, ACE  
N-Series, D/A-Series, V-Series

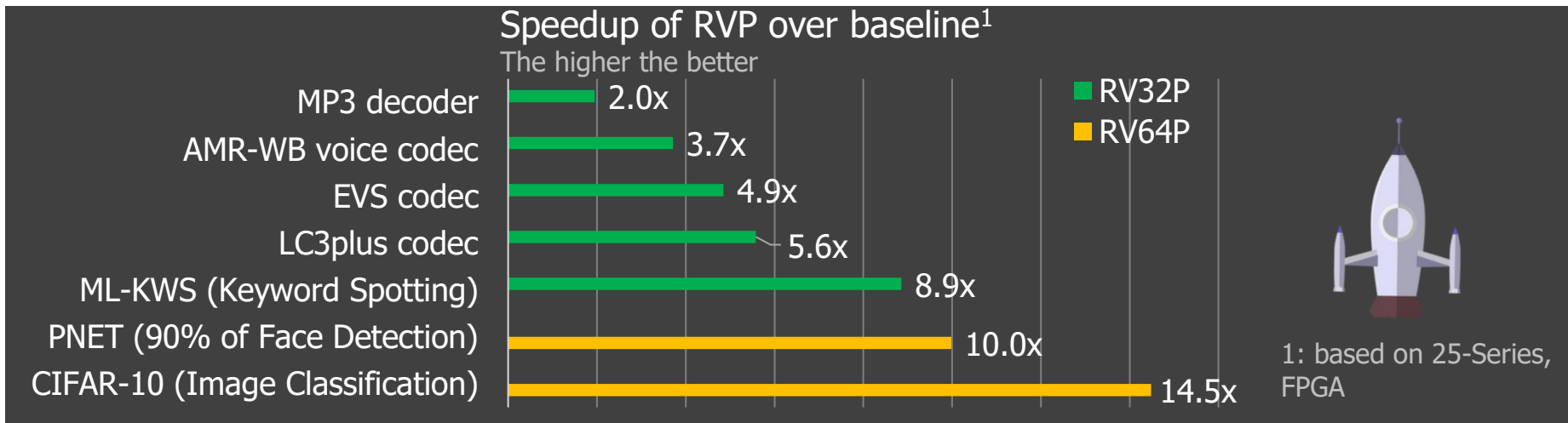


# RISC-V DSP/SIMD Software Solutions

## First implementation of RISC-V DSP/SIMD extension (RVP)



- Compiler automatically generates partial RVP instructions to facilitate development
- **DSP intrinsic functions**: as C-like functions without writing error-prone assembly
- **AndeSoft™ DSP library**
- Increase power efficiency to your DSP applications: **2.0x-5.6x** speedup for codecs, and **8.9x-14.5x** speedup for NN models





# AndeSoft™ DSP Library

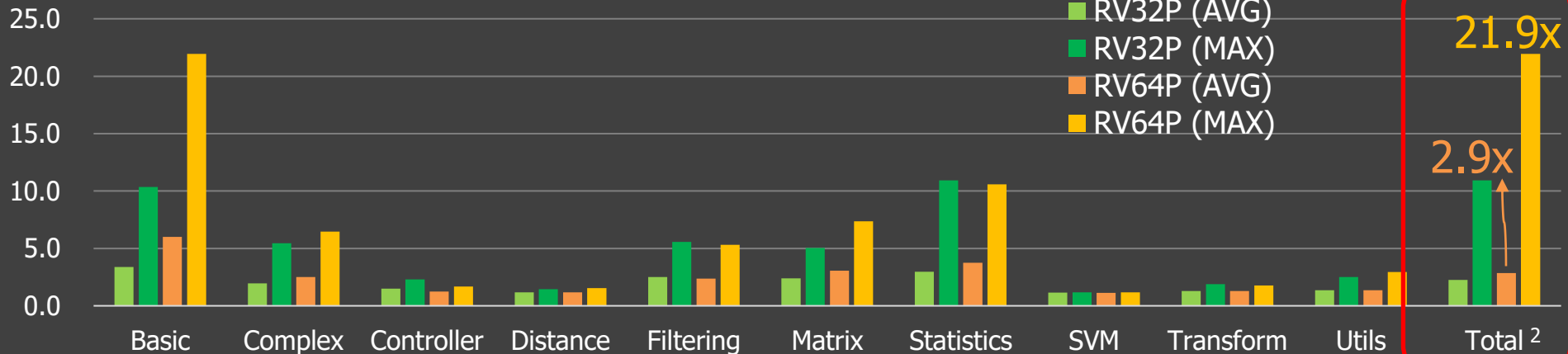


- Optimized low-level DSP functions for **RISC-V Baseline** and **RVP** processors
- **~300 functions in 10 categories**
  - Basic, complex, controller, distance, filtering, matrix, statistic, SVM, transform, utils
- Compatible with CMSIS-DSP API
- Boost signal processing performance
- Speedup **2.9x** in average and **21.9x** in maximum

## Speedup of RVP over baseline<sup>1</sup>

The higher the better

1: based on 25-Series, FPGA



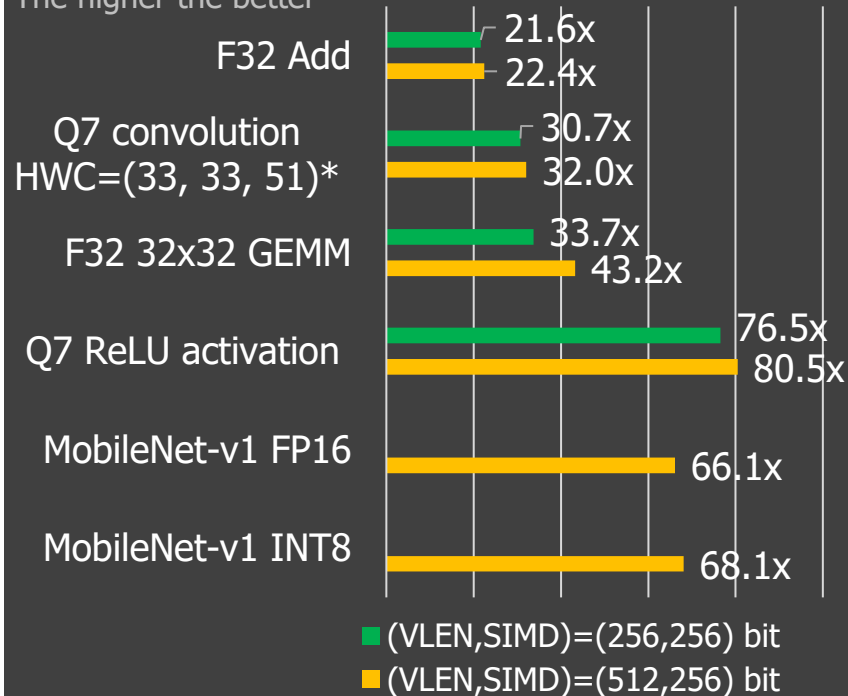
# RISC-V Vector Software Solutions



- Standard software solutions in AndeSight™ IDE
  - AndeSim™: near cycle-accurate simulator
  - Toolchains: intrinsic functions, as C-like functions without writing error-prone assembly
  - AndeSoft™ Vector library:
    - Optimized for RISC-V baseline and RVV CPU
    - > 100 functions in 5 categories: basic, filtering, image, matrix, and transform
    - Compatible with NE10 library APIs
- Advanced software solutions<sup>1</sup>
  - AndeSysC™: SystemC support for AndeSim™
  - AndeClarity™: GUI-based processor pipeline visualizer and analyzer
  - AndeSoft™ NN library: optimized neural network functions

## Speedup of RVV over baseline<sup>2</sup>

The higher the better

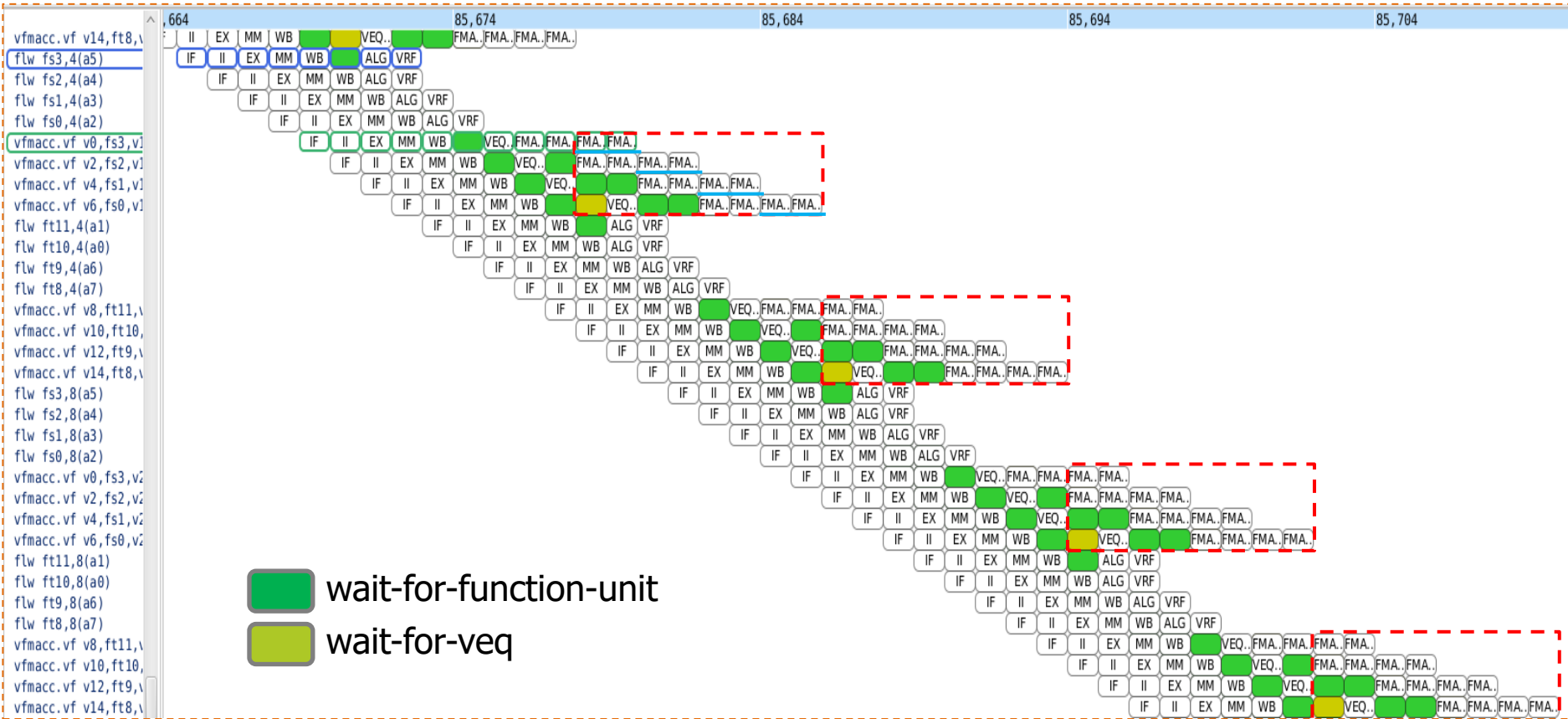


1: separate product packages needed additional licenses

2: based on NX27V, FPGA, 512 KB vector cache

\*: HWC (Height, Width, Channel)

# AndesClarity™ Processor Pipeline Analyzer

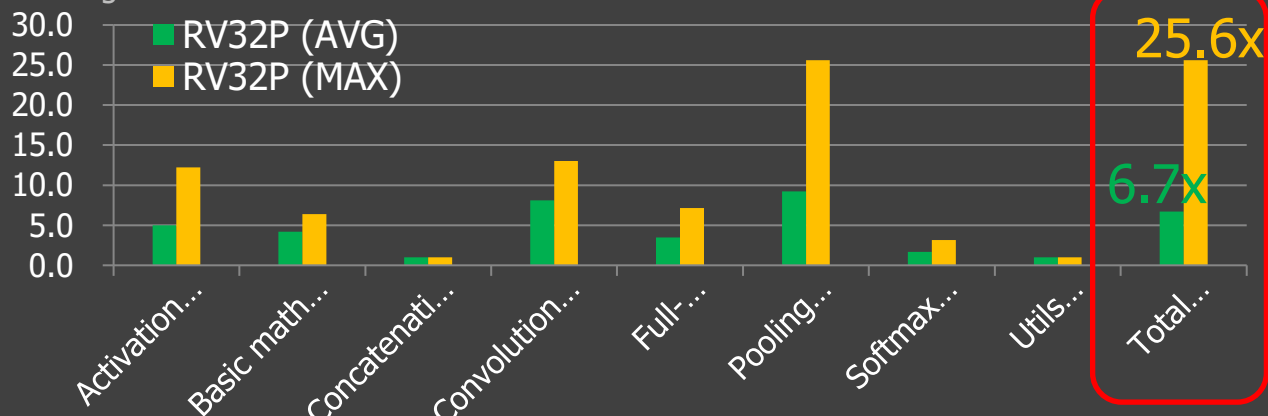


# AndeSoft™ NN Library<sup>1</sup>

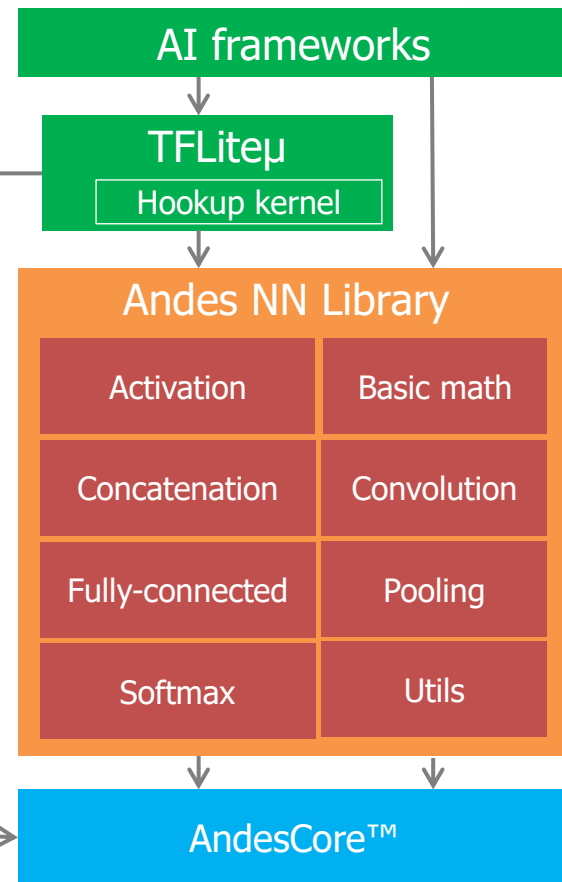
- Optimized neural network functions for **Pure-C**, **RVP**, and **RVV** processors
- Boost NN performance by using SIMD and vector instructions
- **>120 functions in 8 categories**
- Compatible with CMSIS-NN APIs

## Speedup of RVP over baseline\*

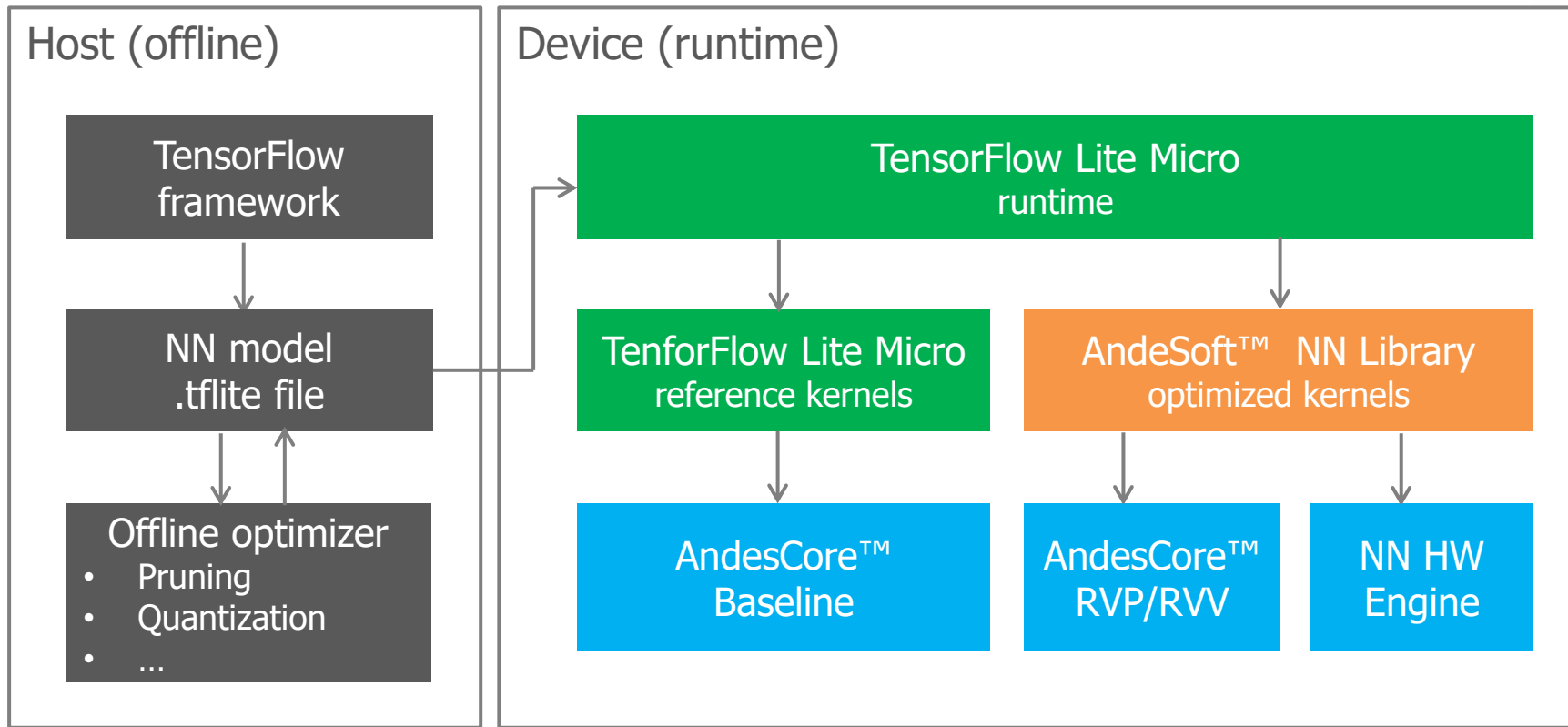
The higher the better



1: separate product packages needed additional licenses



# Inference Flow with TensorFlow Lite Micro<sup>1</sup>



1: available upon request

# AndeSoft™ NN Library API



Categories	Operators	NN Library API <sup>1</sup>
<b>Activation</b>	Sigmoid Tanh	riscv_nn_activate_[s8 s16]
	ReLU Leaky ReLU	riscv_nn_relu_[s8 s16] riscv_nn_leaky_relu_s8
<b>Basic</b>	Element-wise	riscv_nn_ew_[add mul]_s8
<b>Concatenation</b>	Concatenation	riscv_nn_concate_s8_[w x y z]
<b>Convolution</b>	Convolution 2D	riscv_nn_conv_HWC_[IN]_[OUT]_[WEIGHT]_[sft sym asym] <sup>2</sup>
	Pointwise convolution	riscv_nn_conv_1x1_HWC_[IN]_[OUT]_[WEIGHT]_[sft sym asym] <sup>2</sup>
	Depthwise convolution	riscv_nn_conv_dw_HWC_[IN]_[OUT]_[WEIGHT]_[sft sym asym] <sup>2</sup>
<b>Fully-connected</b>	Fully-connected	riscv_nn_fc_[IN]_[OUT]_[WEIGHT]_[sft sym asym] <sup>2</sup>
<b>Pooling</b>	Maximum pooling	riscv_nn_avepool_HWC_s8
	Average pooling	riscv_nn_maxpool_HWC_s8
<b>Softmax</b>	Softmax	riscv_nn_softmax_[u8 s8 s16]
<b>Utils</b>	Reshape	riscv_nn_reshape_s8

1: including but not limited to the list

2: supporting data types of input, output and weight would be either of u8, s8 or s16

# Use Case: CIFAR-10 NN Model

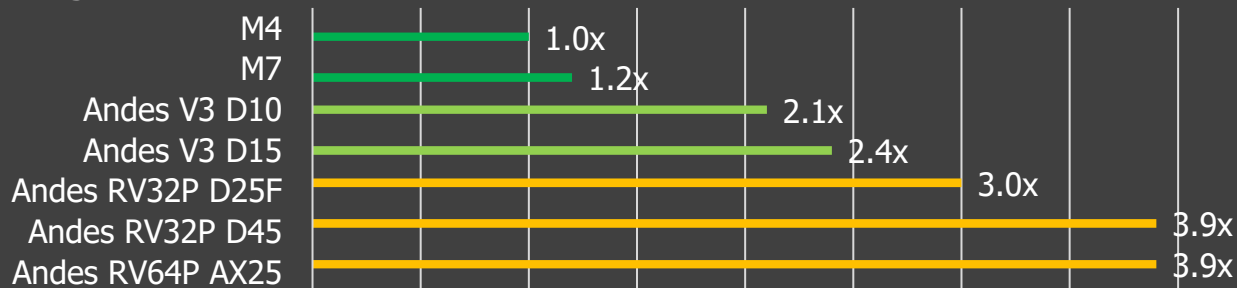


CIFAR-10	Operators	NN Library APIs
Layer 1	Convolution	riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias_fast
	Activation (ReLU)	riscv_nn_relu_s8
	Pooling (maxpool)	riscv_nn_maxpool_HWC_s8
Layer 2	Convolution	riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast
	Activation (ReLU)	riscv_nn_relu_s8
	Pooling (maxpool)	riscv_nn_maxpool_HWC_s8
Layer 3	Convolution	riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast
	Activation (ReLU)	riscv_nn_relu_s8
	Pooling (maxpool)	riscv_nn_maxpool_HWC_s8
Layer 4	Fully-connected	riscv_nn_fc_s8_s8_s8_sft_bias_fast
Layer 5	Softmax	riscv_nn_softmax_s8_fast

- ✓ Performance boost with AndeSoft™ NN library
- ✓ Increase power efficiency
- ✓ Lower response time

## Speedup of CIFAR-10 NN model for image classification<sup>2</sup>

The higher the better



1: based on 25-Series, FPGA

# Use Case: MobileNet-v1 NN Model



## Inference latency of MobileNet-v1 NN model for image classification

The lower the better

MobileNet-v1	Data Type	ISA	VLEN (bit)	SIMD (bit)	Normalized latency <sup>3</sup> (ms @1GHz, 1 core)
CA9 (Xilinx PYNQ) <sup>1</sup>	FP32	NEON		128	703.4
CA53 (Raspberry Pi-3B) <sup>1</sup>		NEON		128	438.5
CA72 (Firefly RK3399) <sup>1</sup>		NEON		128	210.3
CA73 (Kirin 970) <sup>1</sup>		NEON		128	292.4
Andes NX27V <sup>2</sup>	FP16	RVV	512	256	62.3

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1 × 1	94.86%	74.59%
Conv DW 3 × 3	3.06%	1.06%
Conv 3 × 3	1.19%	0.02%
Fully Connected	0.18%	24.33%

知乎 @月臻

1: TVM, <https://github.com/apache/tvm/wiki/Benchmark#arm-cpu>

2: AndeSoft™ NN Library, PyTorchCV imagenet-1k "MobileNet x1.0", <https://pypi.org/project/pytorchcv/>  
 preliminary data based on FPGA and scaling to 1.0 GHz. Real SoC performance will depend on memory subsystem

3: Scale to the same frequency; estimated performance 4 cores="3"\*1 core, the magic number "3" is from experience

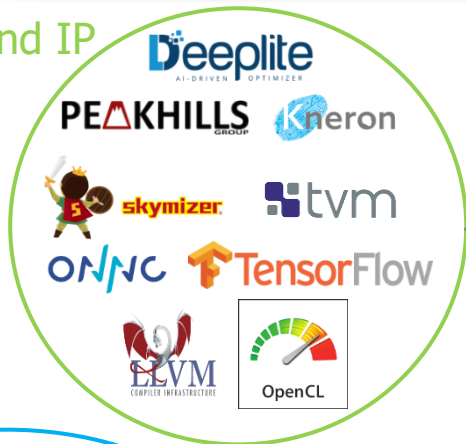




# Andes Ecosystem

# Andes Partners and Ecosystem

AI tools, SW and IP



Security



**ANDES**  
TECHNOLOGY



DSP, Audio and Vision



Development tools



RTOS

# Cyberon DSpotter

- Cyberon DSpotter: Always-listening Voice Trigger/Command
  - DNN-based modeling: higher noise robustness
  - Phoneme-based approach: no need to collect training data
  - Low resource requirement
  - Friendly tool for rapid command customization
  - Global language support: 40+ languages available

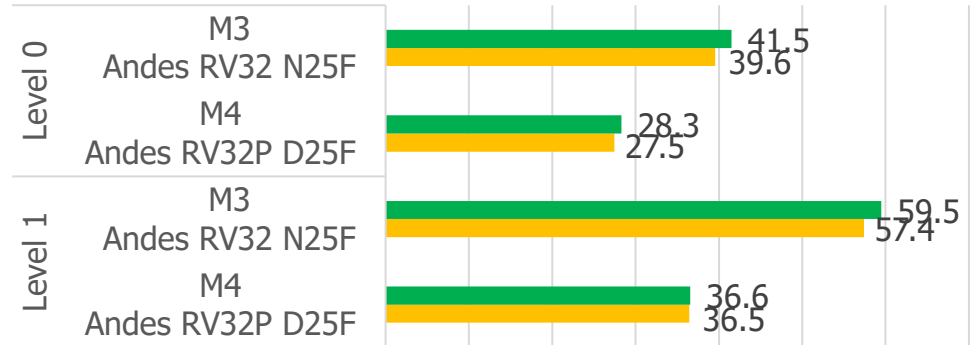


Model	Level 0	Level 1
Hit rate	98.4% (SNR 10dB) 86.5% (SNR 5dB)	98.4% (SNR 10dB) <b>90.4% (SNR 5dB)</b>
Code size	30 KB	30 KB
Data size	95KB + 64B x N	155KB + 64B x N
RAM	40KB + 128B x N	45 KB + 128B x N

N: number of commands

## DSpotter computation (MCPS)<sup>1</sup>

The lower the better

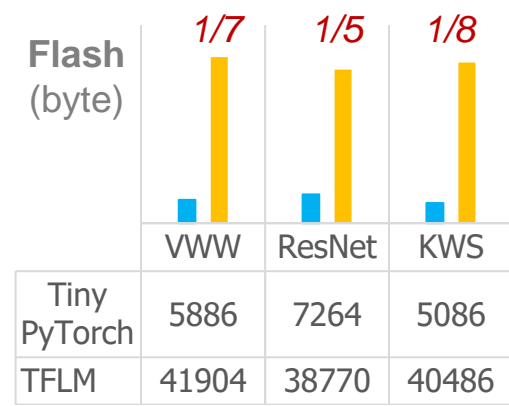
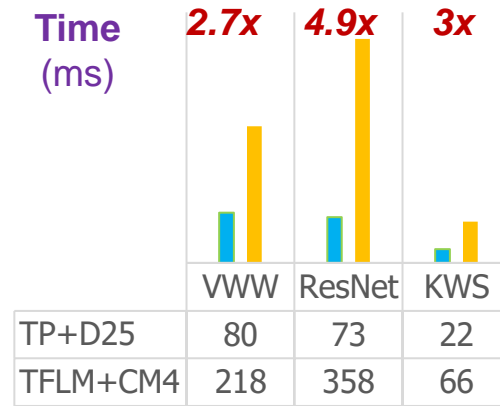
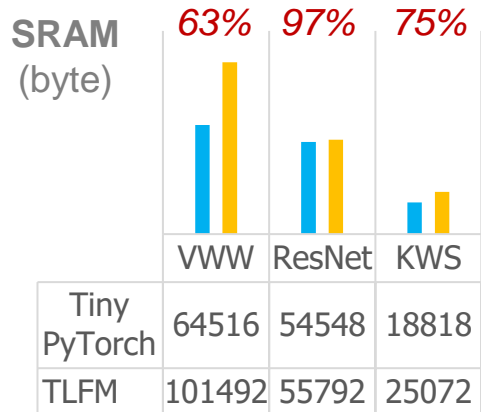


1: "M" data is based on real chip platform; Andes data is based on FPGA with similar configurations

# Tiny ONNC | Tiny and Faster on RISC-V



- **Tiny ONNC** exports popular PyTorch models into a function within *Andes NN Library (libnn)* function calls
- **Tiny** – the transformed *Tiny MLPerf* benchmark uses **only 1/8 flash** and **63% SRAM**, comparing ONNC with *TensorflowLite for Microcontroller (TFLM)*
- **Faster** – on *Andes D25F platform*, the generated benchmark is **4.9x faster** than TFLM on ARM Cortex-M4



# Summary

- AndeSight™ IDE
  - User-friendly and easy-to-use IDE
  - Accelerate your RISC-V software developments with the comprehensive development environment
- AndeSoft™ BSP
  - Highly-optimized toolchains for better performance and smaller memory footprint
  - Well-integrated building blocks to reduce time-to-market
- AI Software Stack
  - Boost the performance for RISC-V AI and IoT applications with optimized AndeSoft™ DSP, Vector and NN libraries

# AndeSight™ IDE Free Download



- **Previous version for evaluation**
- **Three-month time limit**



**Thank You!**