

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

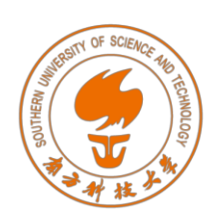
C/C++ Program Design

CS205

Prof. Shiqi Yu (于仕琪)

yusq@sustech.edu.cn

<http://faculty.sustech.edu.cn/yusq/>



南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

C/C++ with ARM



Intel vs ARM



- With the help of C/C++ compilers, C and C++ are platform independent.
- But we need to know some background information on different CPUs.
- Intel achieved a dominant position the personal computer market. But recently ...





ARM

功耗低

ARM CPU 移动设备

arm

- **ARM** (previously an acronym for Advanced RISC Machine and originally Acorn RISC Machine) is a family of reduced instruction set computing (RISC) architectures for computer processors¹.
- ARM is the most widely used instruction set architecture (ISA) and the ISA produced in the largest quantity.



PC 用 ARM CPU

Macbook 新款

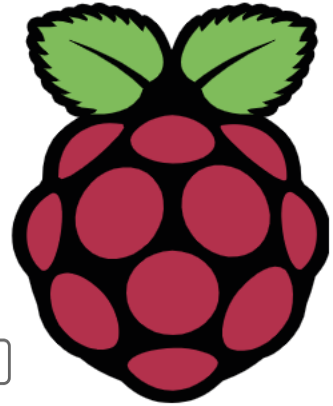
服务器 ARM 服务器

1. https://en.wikipedia.org/wiki/ARM_architecture

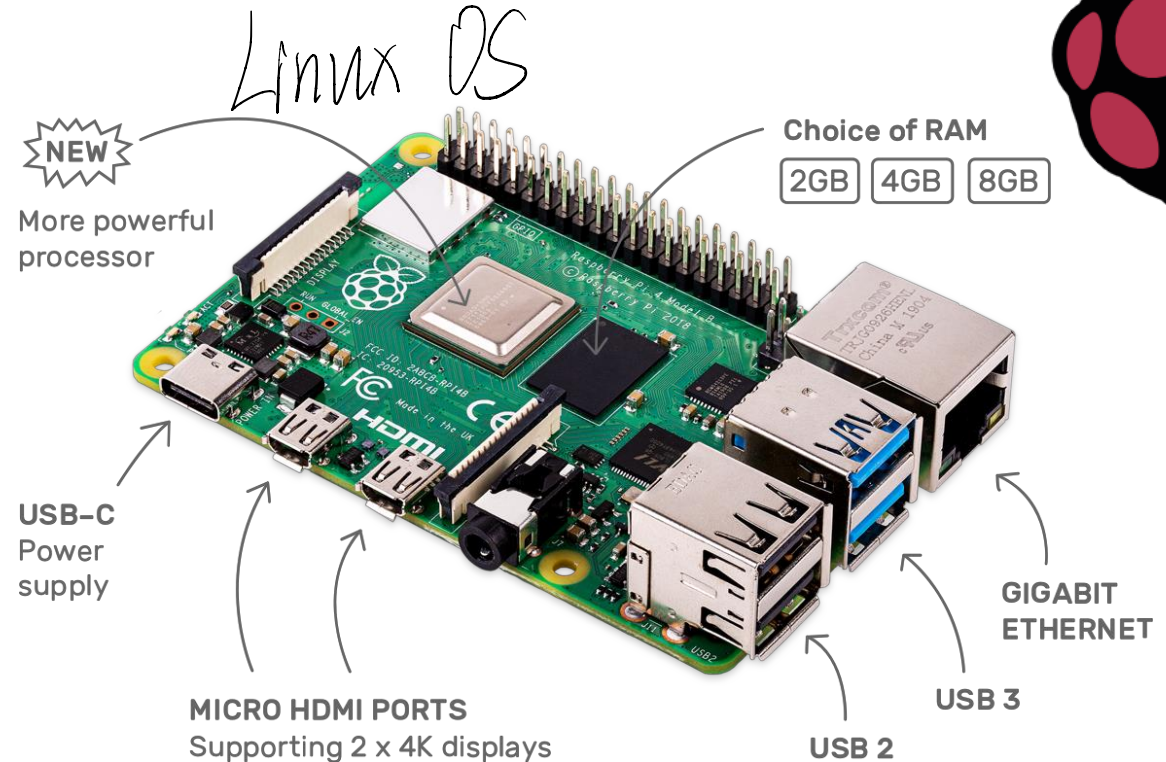


Raspberry Pi 4

<https://www.raspberrypi.org/>



- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- 2 × micro-HDMI ports (up to 4kp60 supported)



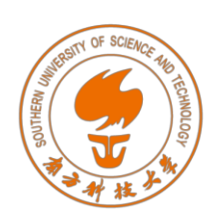


How to develop programs with ARM Development boards

Almost the same with an X86 PC with Linux OS.

- gcc/g++
- Makefile
- cmake

计算能力弱



南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Speedup Your Program



Principle for Programming

简单就是美

Simple is Beautiful !

Short

Simple

Efficient



Some Tips on Optimization

- Choose an appropriate algorithm
- Clear and simple code for the compiler to optimize
- Optimize code for memory
- Do not copy large memory 内存拷贝
- No printf()/cout in loops
- Table lookup (sin(), cos() ...)
- SIMD, OpenMP



An example: libfacedetection

- Face detection and facial landmark detection in **1600 lines** of source code

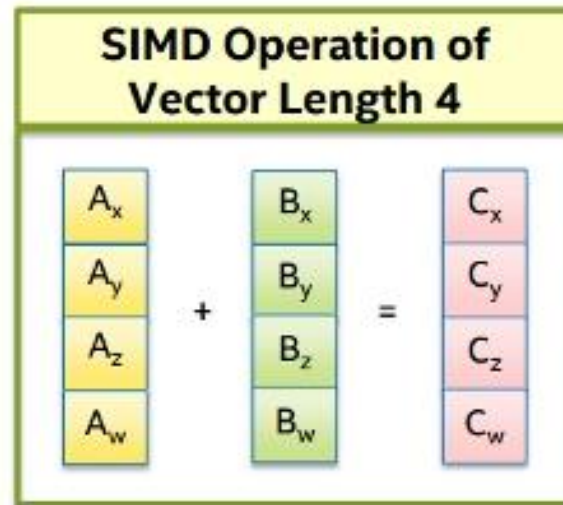
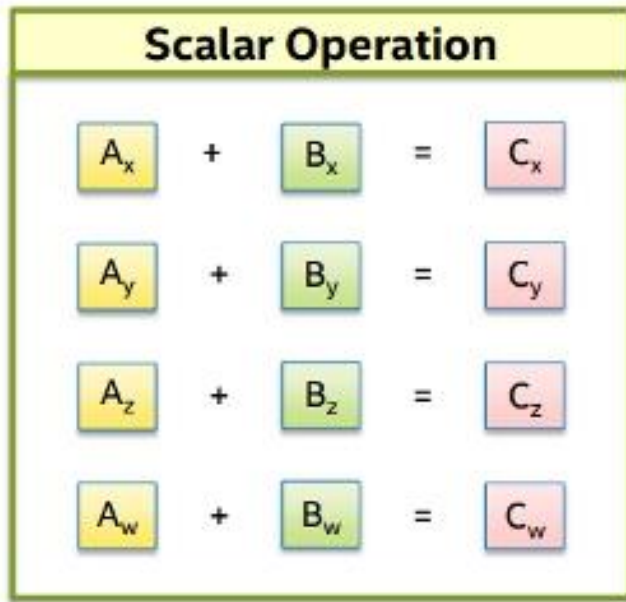
- `facedetectcnn.h`:
 - ✓ 400 lines
 - ✓ CNN APIs
- `facedetectcnn.cpp`:
 - ✓ 900 lines
 - ✓ CNN function definitions
- `facedetectcnn-model.cpp`:
 - ✓ 300 lines
 - ✓ Face detection model
- `facedetectcnn-int8data.cpp`
 - ✓ CNN model parameters in static variables





SIMD: Single instruction, multiple data

SIMD – Single Instruction, Multiple Data



Intel® Architecture currently has SIMD operations of vector length 4, 8, 16

- Intel: MMX, SSE, SSE2, AVX, AVX2, AVX512
- ARM: NEON
- RISC-V: RVV_(RISC-V Vector Extension)



SIMD in OpenCV

- "Universal intrinsics" is a types and functions set intended to simplify vectorization of code on different platforms.
- https://docs.opencv.org/master/df/d91/group__core__hal__intrin.html
- 使用OpenCV中的universal intrinsics为算法提速(1)(2)(3)
 - https://mp.weixin.qq.com/s/_dFQ9lDu-qjd8AaiCxYjcQ
 - https://mp.weixin.qq.com/s/3UmDlImwlQwGX50b1hvz_Zw
 - <https://mp.weixin.qq.com/s/XtV2ZUwDq8sZ8HlzGDRaWA>



OpenMP

#include <omp.h>

Thread 0	Thread 1	Thread 2	Thread 3	Thread 4
i=0-199	i=200-399	i=400-599	i=600-799	i=800-999
a[i]	a[i]	a[i]	a[i]	a[i]
+	+	+	+	+
b[i]	b[i]	b[i]	b[i]	b[i]
=	=	=	=	=
c[i]	c[i]	c[i]	c[i]	c[i]

```
#pragma omp parallel for
for (size_t i = 0; i < n; i++)
{
    c[i] = a[i] + b[i];
}
```



OpenMP

Where should `#pragma` be? The 1st loop or the 2nd?

```
#include <omp.h>
```

```
#pragma omp parallel for
```

```
for (size_t i = 0; i < n; i++)
```

```
{
```

```
    // #pragma omp parallel for
```

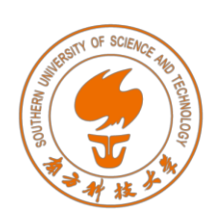
```
    for (size_t j = 0; j < n; j++)
```

```
    {
```

```
        //...
```

```
    }
```

```
}
```



An Example with SIMD and OpenMP



ARM Cloud Server

- Huawei ARM Cloud Server
- Kunpeng 920 (2 cores of many)
- RAM: 3GB
- openEuler Linux



```
(base) yushiqi: ~ $ ssh yushiqi@121.128.128
```

```
Authorized users only. All activities may be monitored and reported.  
yushiqi@121.128.128's password:
```

```
Welcome to Huawei Cloud Service
```

```
Last login: Mon Nov 1 14:06:13 2021 from 116.7.234.238
```

```
Welcome to 4.19.90-2003.4.0.0036.oe1.aarch64
```

```
System information as of time: 2021年 11月 01日 星期一 14:07:44 CST
```

```
System load: 0.00  
Processes: 115  
Memory used: 12.3%  
Swap used: 0.0%  
Usage On: 11%  
IP address: 192.168.0.58  
Users online: 2
```

```
[yushiqi@ecs-01-0002 ~]$ uname -a
```

```
Linux ecs-01-0002 4.19.90-2003.4.0.0036.oe1.aarch64 #1 SMP Mon Mar 23 19:06:43 UTC  
2020 aarch64 aarch64 aarch64 GNU/Linux
```

```
[yushiqi@ecs-01-0002 ~]$ cat /proc/cpuinfo
```

```
processor      : 0  
BogoMIPS      : 200.00  
Features      : fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp  
puid asimdrdm jscvt fcma dcpop asimddp asimdfhm  
CPU implementer : 0x48  
CPU architecture: 8  
CPU variant    : 0x1  
CPU part       : 0xd01  
CPU revision   : 0
```

```
processor      : 1  
BogoMIPS      : 200.00  
Features      : fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp  
puid asimdrdm jscvt fcma dcpop asimddp asimdfhm  
CPU implementer : 0x48  
CPU architecture: 8  
CPU variant    : 0x1  
CPU part       : 0xd01  
CPU revision   : 0
```

```
[yushiqi@ecs-01-0002 ~]$
```



Functions for dot product

```
float dotproduct(const float *p1, const float * p2, size_t n);
```

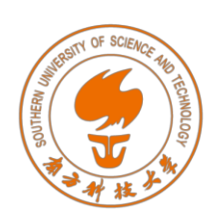
```
float dotproduct_unloop(const float *p1, const float * p2, size_t n);
```

```
float dotproduct_avx2(const float *p1, const float * p2, size_t n);
```

```
float dotproduct_avx2_omp(const float *p1, const float * p2, size_t n);
```

```
float dotproduct_neon(const float *p1, const float * p2, size_t n);
```

```
float dotproduct_neon_omp(const float *p1, const float * p2, size_t n);
```



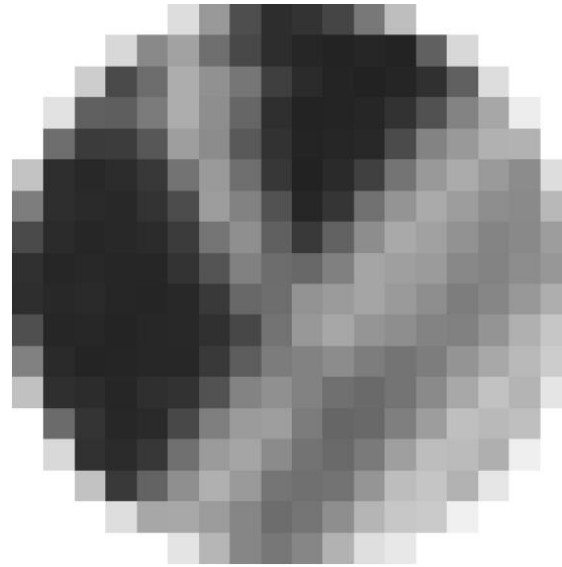
南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Avoid Memory Copy

A trick in OpenCV



What's an image?



$I_{0\ 0}$	$I_{0\ 1}$...	$I_{0\ N-1}$
$I_{1\ 0}$	$I_{1\ 1}$...	$I_{1\ N-1}$
...
$I_{M-1\ 0}$	$I_{M-1\ 1}$...	$I_{M-1\ N-1}$



CvMat struct

...	...
int	flags
int	dims
int	rows
int	cols
uchar*	data
int*	refcount
...	...

modules/core/include/opencv2/core/types_c.h

```
468  typedef struct CvMat
469  {
470      int type;
471      int step;
472
473      /* for internal use only */
474      int* refcount;
475      int hdr_refcount;
476
477      union
478      {
479          uchar* ptr;
480          short* s;
481          int* i;
482          float* fl;
483          double* db;
484      } data;
```



Ref count

Matrix data



step in CvMat struct

- How many bytes for a row of Matrix 4(row)x3(col)?
 - Can be 3, 4, 8, and any other values ≥ 3 .
 - Memory alignment for SIMD



ROI: Region of interest

Mat

Matrix Data

- Mat A

- rows=100
- cols=100
- step=100
- data=0xABCDEF00

A

- Mat B

- rows=100
- cols=100
- step=100
- data=0xABCDEF00

B

- Mat C

- rows=30
- cols=28
- step=100
- data=0xABCE0698

C