
تمرین اول ساختار کامپیوتر

سید محمدرضا جوادی 402105868

دکتر اسدی، پاییز 1403

# Contents

## 1:

الف) روش آن به طور خلاصه بر اساس جمع متوالی است. دقیق تر بگوییم دستگاه برای ضرب دو عدد کوچکتر از 10 "آ" و "ب" ، ب بار آ را با خودش جمع می کند.

( در صورت سوال گفته شده نیازی به بیان نحوه پیاده سازی نیست، ولی خالی از لطف نیست اشاره کنیم در 8 رقم ضرب شونده عدد را قرار می دهیم و در قسمت ضرب کننده عدد ضرب کننده را، حال چرخنده به اندازه عدد ضرب کننده می چرخد و در هر بار عدد ضرب شونده با خودش جمع می شود)

ب) برای بیان بهتر مراحل ضرب شونده را "آ" و ضرب کننده را "ب" می نامیم. همچنین بدیهی است ابتدا ماشین ریست می شود.

مرحله اول: راست ترین رقمِ (کم ارزش ترین، یکان) "ب" مانند بخش قبل در "آ" ضرب می شود و با خروجی (که در ابتدا صفر است) جمع می شود.

مرحله دوم: عدد "آ" به سمت چپ یک شیفت می خورد

مرحله سوم: عدد "ب" به سمت راست یک شیفت می خورد(البته در حقیقت در دستگاه هر دفعه رقم به رقم باید وارد کنیم)

مرحله چهارم:اگر "ب" صفر شد عملیات تمام است، وگرنه به مرحله اول می رویم.

ج) برای بیان بهتر مراحل مقسوم را "آ" و مقسوم علیه را "ب" می نامیم. ایده کلی تفریق متوالی است.

مرحله اول: تفریق "ب" از "آ"

مرحله دوم: اگر نتیجه تفریق مثبت بود خروجی را یک واحد افزایش می دهیم. وگرنه به مرحله قبل باز می گردیم

مرحله سوم: وقتی که حاصل تفریق منفی باشد، عملیات متوقف شده و خروجی ما نتیجه تقسیم است. همینطور در خود ماشین پنجره ای برای نشان دادن عملیات تفریق و جمع وجود دارد، که عددی که در انتها آنجا می ماند باقی  مانده تقسیم است.

## 2.

الف) بی درنگ بودن و قابل اطمینان بودن

ب) توان مصرفی و هزینه

ج)قابلیت اطمینان و بی درنگ بودن

د)کوچک بودن، بی درنگ بودن و قابلیت اطمینان

## 3.

الف) شرایط و محیط به کارگیری این را ایجاب می کند. منابع در محیط مورد استفاده محدود است و در موارد بسیار امکان تعویض مداوم و پشتیبانی لجستیک امکان ندارد. همچنین فضای قرار گیری آن ها هم محدودیت هایی را ایجاد می کند. این نوع کامپیوتر ها به تعداد بسیار زیادی تهیه می شوند و مورد استفاده قرار می گیرند لذا کاهش هزینه ها ضروری است.

ب) نتایج این دستگاه ها بسیاری از مواقع حیاتی است یا اطلاعات با تاخیر نامطلوب در عملیات مورد استفاده کاربردی ندارند. به طور مثال دماسنجی که دما را در گیاه خانه ها اندازه می گیرد برای هشدار دادن اگر دیر اخطار دهد سودی ندارد یا سیستم ترمز اگر دیر تر از موعد عمل کند می تواند مخرب باشد.

ج) در این ماشین ها تنها مهم این است که تا قبل ددلاین نتیجه را برساند و سریع تر از آن لزومی ندارد یا مفید نیست. به طور مثال اگر ددلاین اعلام دما 15 دقیقه باشد برای مصرف کننده فرقی ندارد که در 1 ثانیه دما اعلام شود یا 10 دقیقه.

برای همین به محض بر آورده کردن نیاز های مطلوب، کم کردن هزینه ها و بالا بردن اطمینان پذیری اولویت قرار می گیرد چرا که عملکرد سریع تر و بهتر فرق برای مصرف کننده ندارد و فقط مهم است تا ددلاین تعیین شده نتایج برسد.

## .4

ابتدا برای هر نوع دستور اندازه را مشخص می کنیم.برای کوتاه شدن توضیحات اینجا مشترکا برخی توضیحات را می دهم:

از آنجا که 32 رجیستر داریم برای اشاره به هریک و آدرس دهی 5 بیت نیاز داریم ( 2 به توان 5 = 32) که هر 5 بیت نمایانگر شماره و آدرس یک رجیستر است.

همینطور برای اشاره به هر خانه در هرحافظه به طور مشابه به 10 بیت نیاز داریم ( 1024 خانه حافظه داریم)

پس هرگاه در دستور یک رجیستر نیاز داریم 5 بیت در طول دستور برای آن در نظر می گیریم و هرگاه نیاز به اشاره به خانه ای در حافظه داریم 10 بیت.

### دستورات Arithmetic

4 بیت برای opcode

5 بیت برای ثبات اول، 5 بیت برای ثبات دوم و 5 بیت برای ثبات سوم

در کل 19 بیت برای هر دستور محاسباتی نیاز داریم

| Op:4 | R1:5 | R2:5 | R3:5 |
|---|---|---|---|

### دستورات کار با حافظه

5 بیت برای opcode

5 بیت برای ثبات اول، 5 بیت برای ثبات دوم

10 بیت برای یک خانه از حافظه

در کل25 بیت برای هر دستور نیازمندیم

| Op:5 | R1:5 | R2:5 | Memory:10 |
|---|---|---|---|

### دستورات پرش

4 بیت برای opcode

5 بیت برای ثبات

10 بیت برای یک خانه از حافظه

در کل 19 بیت

| Op:4 | R1:5 | Memory:10 |
|---|---|---|

حال به سوال ها جواب می دهیم:

الف) از آنجا که طول دستورات و همینطور opcode ها برابر است و در مجموع 52 دستور داریم حداقل 6 بیت برای opcode نیاز است پس طول طولانی ترین دستور که حافظه باشد 26 بیت نیاز دارد، با تلف کردن 5 بیت برای دو دستور دیگر طول همه دستور ها را 26 می کنیم.

ب) دستورات حافظه 30 تا هستند، می توانیم با اختصاص دادن 2 مورد باقی مانده به بقیه دستورات همه را در 25 بیت جا کنیم.

طبیعتا 5 بیت اضافه می شود در آن دو مورد ( بر فرض حالت شروع شدن با 11111 و 11110را آن موارد اضافه قرار دهیم) حال 2 * 32 یعنی 64 جای جدید برای دستورات داریم، 10 + 12 تا از قبل اشغال شده ولی می توانیم بقیه را به دستورات محاسباتی اختصاص دهیم.

64-22=42 دستور جدید برای arithmetic داریم.

.5

```
Load Mem[0] // address ebtedaii variable
Store Mem[1], 3 //3 ra dar Mem[1] mirizim
Div Mem[1]
Mul Mem[1]
Store Mem[2], Acc
Load Mem[0]
Sub Mem[2] // to acc mod hast
Store Mem[3],Acc // acc ro rikhtam to khoone 3 hafeze
Store Mem[1], 2
Div Mem[1]
Mul Mem[1]
Store Mem[2], Acc
```

```
Load Mem[3]
Sub Mem[2]
Store Mem[1], 1
Add Mem[1]
Store Mem[0], Acc
```

## .6

فرضیات:

AC = E7 => AC = 1110 0111 = 231 (decimal)

Mem[5] = 7(decimal), Mem[6] = 2(decimal), Mem[7]= 5A = 0101 1010 = 90(decimal)

طبیعتا اگر بعد هر راجع به خانه ای توضیح ندادیم یا تغییری را گزارش نکردیم بلاتغییر است.

توضیح خط به خط:

```
LDA 5
```

خانه  شماره 5 را در ac (انباشتگر) لود می کند و مقدار آن به 7 تغییر می یابد

```
ADD 7
```

ثبات را با مقدارداخل خانه 7 حافظه (90) جمع می کند و در ثبات می ریزد.

7 + 90 =97

مقدار جدید  ac=97یا 0110 0001

```
SHLA
```

ثبات را یک واحد به سمت چپ شیفت می دهد.

0110 0001 =>> 1100 0010

مقدار جدید ثبات 194 است

```
STORE 5
```

مقدار ثبات را در خانه 5 ام حافظه ذخیره می کند

مقدار جدید خانه 5 برابر 194 است.

SUB 6

انباشتگر را با خانه 6 ام حافظه که مقدار 2 دارد جمع می کند.

مقدار جدید ثبات: 196

در انتها مقادیر خواسته شده اینگونه اند:

Ac = 196 , Mem[5] = 194, Mem[6]=2, Mem[7]= 90

## .7

عبارت به صورت پسوندی یا postfix-notation:

a b + f g h - * / c 3 d * - k e j / - / +

```
push a
push b
add
push f
push g
push h
sub
mul
div
push c
push 3
push d
mul
sub
push k
push e
```

```
push j
div
sub
div
add
pop // for returning the result
```

.8

در هر مرحله Aجمله شماره فرد دنباله و B جمله شماره زوج دنباله فیبوناچی است. طبق آرایه داده شده در سوال جملات فرد مثبت و جملات زوج منفی هستند، به همین گونه در هر مرحله ابتدا جمله های بعدی را حساب کرده و جمع می زنیم. توجه کنید که هردفعه 2 جمله بعدی را حساب می کنیم و دوتا دوتا میریم جلو.

حاصل در sum ذخیره می شود.

```
Push 50;
A: 1;
B: 1;
sum: 0;
Label Inja;
Push A; // i
Push sum; // for i +sum
Add; // sum + i
Pop sum; // sum = i + sum
Push B;
Neg; // -i
Push sum; //for -i + sum
Add; // -i + sum
Pop sum; // sum = -i + sum
```

```
Push A;
Push B;
Add;
Push B; // replacing a, b
Pop A; // replacing a, b
Pop B; // b = new value of i
Push A;
Push B;
Add;
Push B; // replacing a, b
Pop A; // replacing a, b
Pop B; // b = new value of i
// now time to count down
Push 1;
Neg;
Add; // alan 50 har dafe yeki kam mishe, har vaght 0 beshe kar tamoome
Jnz Inja;
```

## 9.

### الف)

بنا به اعداد معین شده توسط صورت سوال، نتیجه می گیریم که در کمترین حالت جداگانه، 2 بیت برای دستور بسیار پرکاربرد 3 بیت برای دستورات پرکاربرد و 32 برای کم کاربرد. همینطور برای مشخص کردن ثبات ها 3 بیت می خواهیم و اینکه چگونه پرکاربرد را از کم کاربرد تمیز دهیم در ادامه مطلب شرح داده خواهد شد.

درباره ثبات ها اصلا حساسیتی به خرج نمی دهیم چون احتمال آمدنشان یک دهم است، و به راحتی فاقد اهمیت می شوند. در هرجا ما 3 بیت را برای ثبات اول و 3 بیت را برای ثبات دوم درنظر می گیریم و اگر بنا باشد از بیت های کم کاربرد استفاده شود ما 3 بیت را فرضا 111 در نظر می گیریم و 3 بیت جلوی آن 3 بیت اضافه می کنیم که نمایانگر ثبات کم کاربرد است. به بیان دیگر برای هر آرایش زیر 4 حالت وجود دارد که 6 بیت برای ثبات ها، 9 بیت برای ثبات ها ( اولی کم کاربرد)، 9بیت برای ثبات ها (دومی کم کاربرد) و 12 بیت برای ثبات ها(جفت کم کاربرد) باید اضافه کنیم. وقتی می گوییم پیشفرض منظور حالت اول است، طبیعتا 3 حالت دیگر هم هستند که برای جلوگیری از اطناب هردفعه بیان نخواهند شد و ما اینجا یک بار برای همه شرح دادیم.

حالت اول: 8 بیت پیشفرض

این پرکاربرد ترین حالت است که طولش را مینیمم کردیم. دستور بسیار پر کاربرد 2 بیت اول است ولی مثلا 11 یا هرچیزی را به عنوان نشانه باقی دستور ها می گیریم.

حالت دوم: 11 بیت پیشفرض

2 بیت اول که نشانه بود برای باقی، حال وارد 3 بیت بعدی می شویم و برای دستورات پرکاربرد تبیین شده.باز هم می بینیم از 8حالت1حالت بلا استفاده می ماند، ما امدن این حالت را نشانه ای برای گذر به حالت بعد در نظر می گیریم(فرضا 000یا 111)

حالت سوم: پیشفرض 16بیت

5 بیت اول به ما نشانه دادند که دستور ما از این نوع است. 5 بیت را برای دستورات کم کاربرد اختصاص خواهیم داد. و باقی بیت ها.

توجه: در بالا هم اشاره شد ولی برای تاکید بیشتر است خوب است یاد آوری کنیم که کامپیوتر برای تمیز دستورات از الگوریتم زیر پیروی می کند:

فرض که نشانه عبور کردن 1 های متوالی باشد،

اگربا 11111شروع شود نوع آخر است، وگر با 11نوع دوم، و در غیر اینصورت نوع اول.

پس از تعیین نوع دستور و خوانش دستور، برای ثبات ها اگر 111 بود 3 بیت بعدی شماره ثبات کم کاربرد در غیر اینصورت این 3 بیت برای پرکاربرد است و به بقیه دستور می پردازد.

ب)

حداقل می شود پیشفرض دستور اول که 8 بیت است و حداکثر بدترین حالت دستور آخر است که 10 + 12 یعنی 22 بیت است.

برای میانگین احتمال آمدن هریک در تعداد بیت ها ضرب می شود که بیان دیگر امید ریاضی است، توجه کنید آمدن دستور از ثبات و همینطور ثبات اول از دوم مستقل است پس برای اشتراک در هم ضرب می شوند. جواب دقیقا 9.5544 است محاسبات:

فرمت اول در 4 حالت:

$$0.9 * 0.9 * 0.8 * 8 + (0.9 * 0.1 * 0.8 * 11) *2 + 0.1 * 0.1 * 0.8 * 14$$

فرمت دوم در 4 حالت:

$$0.9 * 0.9 * 0.16 * 11 + (0.9 * 0.1 * 0.16 * 14) *2 + 0.1 * 0.1 * 0.16 * 17$$

فرمت سوم در 4 حالت:

$$0.9 * 0.9 * 0.04 * 16 + (0.9 * 0.1 * 0.04 * 19) *2 + 0.1 * 0.1 * 0.04 * 22$$

که نتیجه جمع 3 محاسبه بالا می شود 9.5544

ج)

در حالت ثابت، 14 بیت نیازمندیم از برای این که کلهم 3 + 7 + 32 دستور داریم که 6 بیت نیازمندیم، و کلا 15 ثبات داریم و برای هریک 4تا. پس 4 + 4 + 6 به ما عدد 14 را نتیجه می دهد.کاهش ما برابر عدد زیر است:

$$(9.5544 / 14) * 100 = 68.24 \rightarrow 31.76\% \text{ reduction}$$

## .10

الف)

```
PUSH A0
PUSH A1
ADD
PUSH A2
PUSH A3
SUB
MUL
PUSH A0
ADD
POP D
```

ب)

```
LOAD A0
ADD A1
STR D
LOAD A3
NEG
ADD A2
MUL D
ADD A0
STR D
```

## .11

الف)

بدیهتا برای تعیین دستورات ۴ بیت نیازمندیم همانطور که در سوال ذکر شده که ۱۶ دستور داریم. همینطور برای هر ثبات ۵ بیت. حال با این فرضیات به هر فرمت می پردازیم، توجه کنید مجموع تعداد بیت ها در هردو فرمت یکسان و برابر ۱۴ است:

فرمت اول ۴ بیت اول را به opcodeاختصاص می دهد و 5 بیت را به ثبات، پس 5 بیت باقی مانده برای مقدار immediateباقی می ماند.

فرمت دوم 4 بیت به opcode 5 بیت به ثبات اول و 5 بیت به ثبات دوم اختصاص می دهد.

ب)

راهکار:

با فرض داشتن همین دو فرمت و صرف اختصاص بیت های بیشتر در فرمت اول به یک بیت در اول دستور نیازمندیم تا فرمت را شناسایی کند، اگر از اولی بود(مثلا بیت اول ۰ بود) می داند طول آن چه قدر است( مثلا ۱۷ یا... ولی همیشه فرمت اول همین خواهد بود) و گر از دومی (مثلا بیت اول یک بود) می داند از فرمت دوم است و ۱۵ بیتی است. البته شایان ذکر است که در صورت تخصیص بیت ها به صورت متفاوت به immediateها مختلف ناچاریم بیش از یک بیت را به این مهم اختصاص دهیم، مانند آنچه در سوال 9 رخ داد، و پیچیدگی هایی در فرایند decodingخواهد داشت.

معایب و پیچیدگی های هنگام پیاده سازی:

از موارد مشکل ساز به پایین آمدن عملکرد و سرعت پردازش می توان اشاره کرد. این نوع طراحی زمان و انرژی بیشتری از پردازنده می گیرد چه را که علاوه بر انجام دستور ها باید قسمتی را نیز به صرف کدگشایی دستور کند و تشخیص دهد 0 و1 هارا چگونه بخواند. همچنین این افزایش پیچیدگی باعث می شود تا در هر کلاک دستورات کمتری انجام شودو سرعت پایین بیاید، طبیعتا این نوع طول دستور متغیر به هدف استفاده کمتر از حافظه ایجاد شد نه افزایش کارآمدی و سرعت. همچنین درلایه های پایین تر و لایه های سخت افزاری پیچیدگی های بیشتری برای طراحی پردازنده بوجود می آید و منجر می شود نسبت به معماری risc زمان بیشتری صرف تولید توسعه و بهبود آن شود.

## .12

۲کیلو بایت می شود ۲۰۴۸بایت و برای آدرس دادن به هر بایت ما ۱۱ بیت نیازمندیم. از آنجا که ۸ ثبات داریم ۳ بیت هم برای آدرس دهی آن ها نیازمندیم. با این فرضیات به سراغ دو بخش سوال میرویم.

آ)

با توجه به اینکه هر سه فرمت ۱۰ دستور دارند در کل ۳۰ دستور داریم، که در ۵ بیت اول جا می دهیم. دستور فرمت اول ۹ بیت برای ثبات ها، فرمت دوم ۳بیت برای ثبات و ۸ بیت برای داده (کلا ۱۱بیت) و فرمت سوم هم ۱۱ بیت برای اشاره به حافظه نیاز دارد.برای اینکه همه دستورات در یک طول جا شوند حداقل علاوه بر ۵ بیت، ۱۱ بیت هم نیازمندیم. در کل طول دستورات می شود ۱۶.

۱۶ بیت یعنی ۲ بایت و ما ۲۰۴۸ بایت داریم در حافظه پس می توانیم ۱۰۲۴ دستور را جا دهیم.

ب)

توجه* با توجه به احتمال های هر یک از دستورات و ثبات ها جواب می تواند متفاوت باشد. ما اینجا به دلیل عدم بیان برتری یکی بردیگری همه احتمالات را برابر می گیریم و فقط بر طول دستورات تمرکز می کنیم.

برای حداکثر کردن تعداد دستورات میانگین را حداقل می کنیم. متاسفانه هر فرمت ۱۰ دستور دارد که ناچار حداقل ۴ بیت می خواهد، هیچ گونه جدا کردنی هم بدرد ما نمی خورد چون احتمال ها برابر است فقط کار سخت تر می شود و دستورات طولانی تر، بهترین کار همان تخصیص ۵بیت به opcodeاست، که همه در آن 5 بیت دستورات را قرار می دهند، صرفا هنگامی که فرمت اول باشد طول دستور14 و فرمت دوم و سوم برابر 16 است(این که کدام دستور است و طول دستور چقدر است را از opcodeمتوجه می شود) یعنی صرفا بیت های بلا استفاده در حالت قبلی را حذف می کنیم. میانگین این سه عدد می شود 46 تقسیم بر 3 بیت، با محاسبات زیر داریم:

2048 / (46/3 * 8) = 1068.52

که اگر رو به پایین تقریب بزنیم می شود 1068 تا دستور(در محاسبه من توجه داشته باشد 8 با رنگ قرمز به این دلیل است که تعداد بیت را به بایت تبدیل کنیم)

## .13

## آ)

### فرمت دستور عمل ها

با وجود اینکه arm یک معماری risc هست ولی به صورت ترکیبی از دستورات 16 بیت و 32 بیت استفاده می کند تا به شکل پویایی هم سرعت را بالا ببرد و هم استفاده حافظه را بهینه کند. شکل های اصلی دستورات از جنس محاسبات و عملیات بر روی رجیستر ها و همچنین load و store است

در آنسو پردازنده های mips هم کاملا riscهستند و همه دستورات 32 بیتی ( یا در برخی نسخه های آن تماما 64بیتی). همانطور که در لکچر درس داشتیم دستورات به سه فرمت و برای سه نوع کار تقسیم می شوند :

R format → register orders

دستورات با کار با ثبات ها

I format →immediate and register orders

دستورات کار با مقدار عددی و ثبات

J format → jump order

دستورات پرش

### مدل های دسترسی به حافظه(store/load)

در این مورد هردو مشترکا از load/store یا همان register registerاست که باید با ثبات ها کار کند، به این شکل که محاسبات با ثبات هست و برای برداشتن مقداری از حافظه یا ریختن مقداری در حافظه از دستورات loadو store استفاده کنیم و با ثبات ها عملیات نوشتن و خواندن را انجام دهیم. بنابراین نمی شود مستقیما به حافظه دسترسی پیدا کرد.

(البته در armهای متفاوت برخی دستورات خیلی خاص و محدود برای دسترسی مستقیم به حافظه و انجام عملیات وجود دارد ولی باز هم به صورت کلی یک وضعیت را دارند)

### استفاده از ثبات ها

ARMSاز 16 ثبات استفاده می کند و کاربرد دقیق هریک عبارت است از:

- General purpose registers: R0, R1,R2,3,R4,R5,R6,R7,R8,R9,R10,R11,R12 رجیستر های عمومی

- Link register: LR or R13لینک رجیستر ها
- Stack pointer: SP or R14رجیستر اشاره گر به استک
- Program Counter: PC or R15رجیستر شمارشگر برنامه

ولی MIPSاز 32 رجیستر عمومی استفاده می کند همچنین رجیستر هایی با کاربرد خاص دارد مانند رجیستر عدد ثابت صفر یا ثبات های اشاره گر به پشته.

### طول دستور العمل

در MIPSفیکس 32 بیت است. اما در ARM دستورات به منظور بهبود استفاده از حافظه دارای دستورات 16(thumb) بیتی و 32 بیتی است. البته توجه داشته باشید 16 بیتی اغلب موارد استفاده در بهبود کامپایلر ها دارند وگرنه برای اکثر دستورات اسمبلی مانند 32 mips بیتی است. پس می توان گفت طول دستور در میپس ثابت و آرم متغیر است.

### ب)

Mips مدت مدیدی پرفروش ترین پردازنده بود به دلیل کاربرد ویژه ای که در سیستم های نهفته دارد که تا امروز به قوت خود باقی است. از جمله این سیستم ها می توان به روتر ها، ps2 و ps1،psp و همچنین در تلوزیون ها در بخش گیرنده های تلویزیونی ( بخصوص توسط شرکت برودکم) اشاره کرد. دلیل این محبوبیت در کامپیوتر های نهفته اندازه کوچک، هزینه کم بخاطر سادگی و قدرت پردازش مناسب است.

در آنسو پردازنده های Armجدیدا با پیشرفت تکنولوژی مانند پدید آمدن تبلت ها، گوشی ها و همینطور اینترنت اشیاء محبوب است.دلیل این موضوع ببیشتر در توان مصرفی و پردازش بالاست، اینکه در محیط هایی که انرژی کم است به چنین پردازنده کم مصرفی محتاجیم. مثال های معروفی که می توان بیان کرد محصولات شرکت اپل است، به خصوص در پردازنده های خود از این معماری استفاده می کند و بسیاری از مک بوک ها، آیپد ها و همینطور آیفون ها از این معماری بهره می برند. البته فقط اپل نیست و شرکت های دیگر مانند گوگل و سامسونگ هم در تولید گوشی ها به این معماری علاقه مندند. به دلیل مصرف کم و عملکرد مناسب این معماری در صنعت اینترنت اشیا و هوشمند سازی کاربرد عمده ای دارد. جالب است بدانید پردازنده محبوب رازبریپای نیز از این نوع معماری استفاده می کند.

### ج)

ARMS

نکات مثبت آن به مصرف کم انرژی و حافظه و در عین حال قدرت بالای پردازش است. در سیستم های نهفته ای که مسئله انرژی اولویت اول است این معماری خودنمایی می کند.

برای نکته منفی آن می توان از پیچیده شدن کدزنی برای آن و طراحی پیچیده تر نسبت به MIPSاشاره کرد.

MIPS

نقطه قوت آن در سادگی و قیمت است. همچنین قدرت بالا و همینطور ثابت بودن طول دستور ها نیز مزیتی دوچندان برای آن است. در هرجا که قیمت، سادگی سخت افزار و استفاده راحت(کد زدن) مطرح باشد این پردازنده هم مطرح است.

در مقابل اما مصرف انرژی بیشتر نسبت به رقیب خود نکته منفی محسوب می شود.