

# RX23T Inverter Kit – What enables you to use Float in your motion control algorithm?

## Introduction

The new microcontroller families RX23T and RX24T featuring the new RX v2 core are designed to ensure the best compromise in terms of performance and cost to address motion control. In case of the RX23T, the latest benchmarks are showing a performance of 80 DMIPs at 40 MHz. In case of the RX24T, the MCU reaches up to 160 DMIPs at 80 MHz. For both cores, according to EEMBC Certified Scores, the RX23T/24T deliver up to 4.25 CoreMark per MHz. For more details, please refer to the official website:

[www.eembc.org/coremark/](http://www.eembc.org/coremark/)

These MCU's are designed to operate at 5V to ensure the highest noise immunity and incorporate key hardware modules like the Floating Point Unit (FPU). The FPU is the main topic of this paper as it is becoming fundamental for software engineers to deliver state of the art software, easy to maintain and to test.

## Why has Renesas implemented a Floating Point unit on the latest RX23T/RX24T families?

The current RX62T & RX63T have been extremely successful in the current generation of motor control solutions. These families already integrate Floating Point Unit. Renesas decided to improve the overall RX Core and the FPU was one component of it.

Please find below the instructions of the RX V2 Core included in "red" the new one added to the V1.

  New instructions of the RX V2 Core

Arithmetic/Logic			Data Transfer				DSP			
ABS	MAX	RORC	MOV	POP	PUSHC	SCCnd	EMACA	MSBLO	MVTACGU	
ADC	MIN	ROTL	MOVCO	PUSH	PUSHM	STNZ	EMSBA	MULHI	MVTACHI	
ADD	MUL	ROTR	MOVLI	POPC	REVL	STZ	EMULA	MULLH	MVTACLO	
AND	NEG	SAT	MOVU	POPM	REVV	XCHG	MACHI	MULLO	RACL	
CMP	NOP	SATR	System manipulation		Branch	Strings	MACLH	MVFACGU	RACW	
DIV	NOT	SBB	BRK	MVTC	Bcnd	SCMPU	MACLO	MVFACHI	RDACL	
DIVU	OR	SHAR	CLRPSW	RTE	BRA	SMOVB	MSBHI	MVFACLO	RDACW	
EMUL	RMPA	SHLL	INT	RTFI	BSR	SMOVF	MSBLH	MVFACMI	FTOI	
EMULU	ROLC	SHLR	MVTIPL	SETPSW	JMP	SMOVU	Floating-point		FTOU	
SUB	TST	XOR	MVFC	WAIT	JSR	SSTR	FADD	FMUL	ITOF	
Bit manipulation						RTS	SUNTIL	FCMP	FSUB	ROUND
BCLR	BMcnd	BNOT	BSET	BTST	RTSD	SWHILE	FDIV	FSQRT	UTOF	

Based on the three additional FPU instructions below and the fourteen DSP new instructions, the RX v2 core offers higher calculation accuracy by executing 32-bit multiply by a single instruction. Furthermore, the specific instructions to manage the accumulator and the rounding processing ensure high

overall performance. The new RXv2 offers up to 2 DMIPs per MHz, compared to 1.65 DMIPs per MHz of the RX v1.

New RX v2 Instruction	Function
FSQRT	Floating point square root
FTOU	Floating point to integer conversion
UTOF	Integer to floating-point conversion

By comparing the implementation of a three shunts sensorless vector control algorithm on the same Microcontroller RX23T, the on-chip FPU module shows high benefits by decreasing the processing time by 27% and by reducing the CPU load by 28%.

RX23T			
CPU clock	40 MHz, V2 core		Perform. Diff.
Arithmetic used in software	Fixed	Float	
Control loop Timing	51 µs	40 µs	27%
Code size in Flash	26 KB	20 KB	30%
Code size in RAM	3 KB	2 KB	50%
CPU load @ 16 KHz PWM/Control loop	82%	64%	28%

The measurements above are done on the inverter kit software running on the RX23T kit called YROTATE-IT-RX23T kit. The software incorporates the following software blocks:

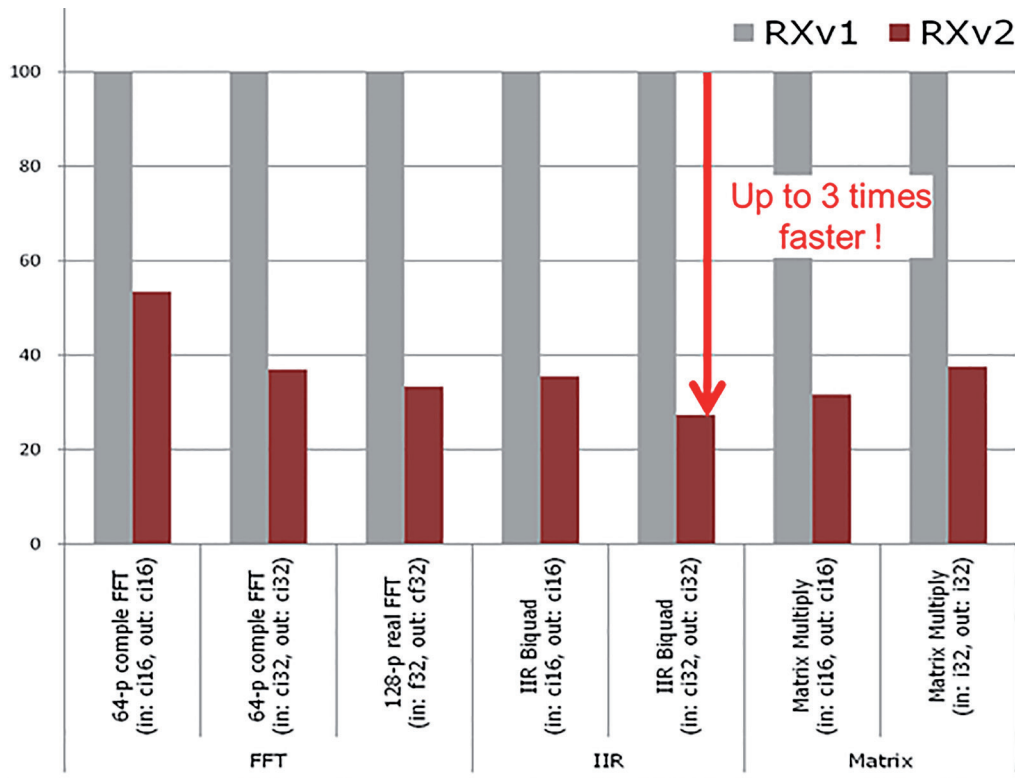
- Proportional-Integral Current Controller
- Proportional-Integral Speed Controller
- Clamped Pulse Width Modulation
- Clarke and Park Transformations
- Flux Phase estimator
- Speed estimation
- Three shunt current reading

The complete source code of the software used for the benchmarks and tests are fully available and royalty-free.

Additionally, in many motion control applications, the

equipment is connected to sensors and filtering the signals and data coming from them is a fundamental tasks. That's why Renesas performed some comparisons between the RX V1 and V2 versions, in various filtering operations.

## RX23T Inverter Kit – What enables you to use Float in your motion control algorithm?



The complete material related to the RX23T kit available on the website:

[www.renesas.eu/motorcontrol](http://www.renesas.eu/motorcontrol)

First, let's dive into the embedded software source code of the inverter. The constants used in the inverter algorithm are showing high resolution and accuracy. It means the overall control of the torque and speed is accurate. The development and the maintenance of the embedded software is becoming easier. Furthermore, the variables used in the firmware are directly representing the real units in Ampere, in Volt, in Hertz, in Weber and in Henry, etc.

Using Float variables is a

As you can observe above, for each filtering operations using Fast Fourier Transform (FFT), for the Infinite Impulse Response (IIR) and the Matrix multiplication, the newest core RX v2 is minimum two times faster. The graph is normalized compared to the RX V1 core, the data are relative to the RXv 1 core. There is no timing unit.

In case of the second order filter (IIR Biquad), the RX v2 core is three times faster than the RX v1 core thanks to the new instructions, at the same clock frequency.

Up to now, it was presented the reasons to embed a Floating Point Unit module and the results in terms of performance, code size and overall code efficiency.

### How can I move from fixed arithmetic to float?

The first step is to start by evaluating available 3-phase inverter reference platform driving already Permanent Magnet Motors. The kit based on RX23T (e.g. YROTATE-IT-RX23T) enables any developer to quickly experiment the usage of float variables.

```
// floating point numerical constants
#ifndef F_CONST
#define F_CONST
#define PI (3.141592654) // 180
#define PISIXTH (PI / 6.0) // 030
#define PIPOURTHS (PI / 4.0) // 045
#define PITHIRDS (PI / 3.0) // 060
#define PIHALVES (PI / 2.0) // 090
#define TWOPIHIRDS (PI * 2.0 / 3.0) // 120
#define THREEPIFOURTHS (PI * 3.0 / 4.0) // 135
#define FIVEPISIXTH (PI * 5.0 / 6.0) // 150
#define SEVENPISIXTH (PI * 7.0 / 6.0) // 210
#define FIVEPIFOURTHS (PI * 5.0 / 4.0) // 225
#define FOURPIHIRDS (PI * 4.0 / 3.0) // 240
#define THREEPIHALVES (PI * 3.0 / 2.0) // 270
#define FIVEPIHIRDS (PI * 5.0 / 3.0) // 300
#endif
```

way to represent real numbers on a Microcontroller.

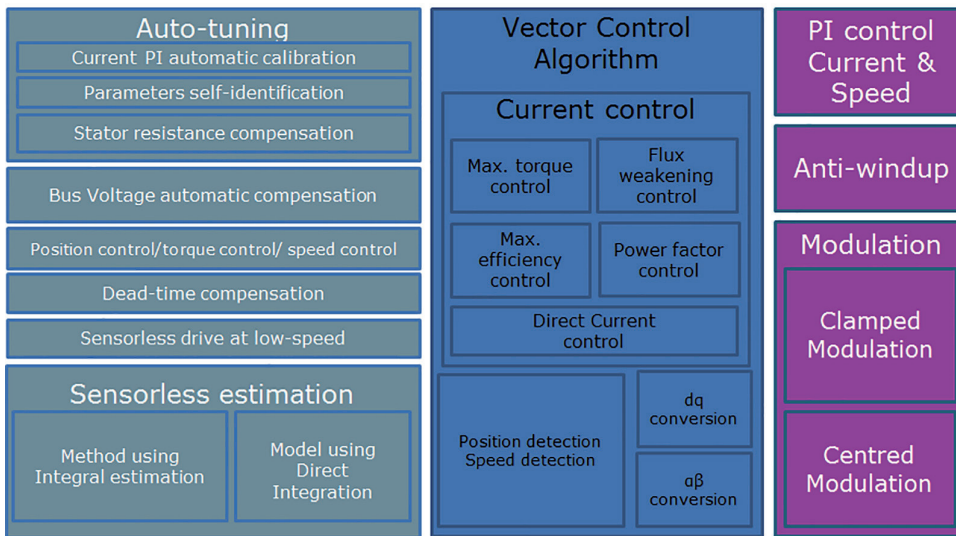
```
float32_t
ium_off, // IU analog channel offset
ivm_off, // IV analog channel offset
iwm_off, // IW analog channel offset
r_sta, // stator resistance
l_syn, // synchronous inductance
pm_flx, // permanent magnets flux
fb_gain, // flux amplitude feedback gain
flx_lpco_hz, // approx. flux estimation filter cutoff frequency
c_poli, // number of polar couples
i_start, // startup current (peak)
is_rdw, // startup current decreasing rate
vbus, // bus voltage
vbus_minf, // filtered available bus voltage (minimum value)
i_max, // maximum total current
id_max, // maximum d current
iq_max, // maximum q current
ibr, // beta current reference
rpmrif_x, // reference speed (ramp input) [rpm]
rpmrif_y, // reference speed (ramp output) [rpm]
r_acc, // acceleration ramp [rpm/main_loop_duration]
r_dec, // deceleration ramp [rpm/main_loop_duration]
rpm_max, // maximum speed [rpm]
rpm_min, // minimum speed [rpm]
min_speed, // minimum speed [rad/s]
max_speed, // maximum speed [rad/s]
omegae, // electrical angular speed
omf, // electrical angular speed (filtered)
maxerr, // maximum speed error [rad/s]
```

All the internal representation of physical quantities of the system of the 3-phase inverter is very simple using floating point. There is no need to normalize the variables or scale them. Thanks to these the code is becoming easy to manage and read. The motor model can be easily worked out without scaling efforts on each intrinsic.

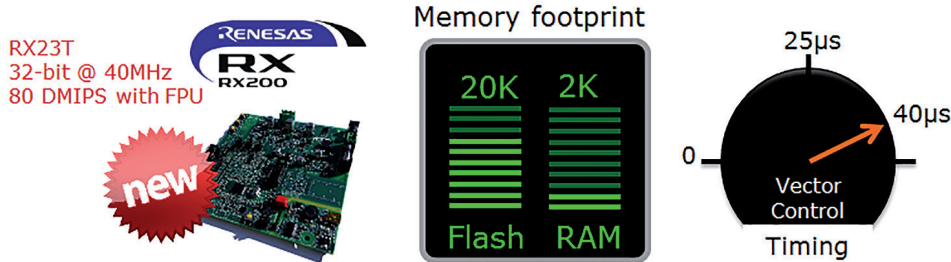
Furthermore, let's have a look at the "Flux Phase Estimation" block as shown on the next page.



## RX23T Inverter Kit – What enables you to use Float in your motion control algorithm?



By setting up the RX compiler optimization level to “Max”, the performances below are reached.

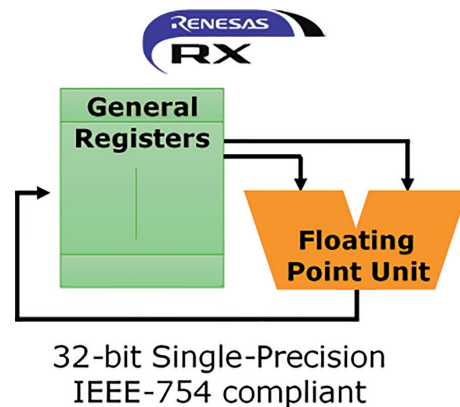


The resources used by the software running on the RX23T are very limited: 20 KB flash and 2 KB RAM, so it means, up to 108 KB free for the application on the RX23T with 128 KB flash.

### Conclusion

This paper is describing in details how much the performance of the inverter algorithm are improved using the RX23T Floating Point Unit.

The overall code is reduced, the overall software implementations are executed much faster as no scaling or saturation occurred. Furthermore, it offers a higher accuracy than the integer implementation as the real physical units are used in the embedded software. Finally, the FPU guarantees a faster computation time for PI controllers



and estimators. So, stop waiting and download the latest material on [www.renesas.eu/motorcontrol](http://www.renesas.eu/motorcontrol) to evaluate by yourself the source code delivered royalty-free.

---

Before purchasing or using any Renesas Electronics products listed herein, please refer to the latest product manual and/or data sheet in advance.

---