AN12424

SPT Programming on S32R

Rev. 0 — May 2019 Application Note

by: NXP Semiconductors

Contents

1 Introduction

The S32R274 is a 32-bit Power Architecture® based Microcontroller (MCU) unit for automotive applications. It extends the MPC5775K family by a value device optimized for surround RADAR sensors and midrange front RADAR sensors. The family members are designed to address advanced RADAR signal processing capabilities and merge it with microcontroller capabilities for generic software tasks and car bus interfacing.

S32R274 meets the high performance computation demands required by modern beamforming fast chirp modulation RADAR systems by offering unique signal processing acceleration together with powerful multi-core architecture. The S32R274 supports automotive safety applications that require a high Safety Integrity Level (ASIL) and adds SHE compliant security features to prevent unauthorized manipulations. The high integration level of S32R274 enables the customer to build compact, safe and secure, low cost RADAR sensors with leading edge performance. The family members have a high level of software compatible for shared hardware modules.

S32R274 is a high end automotive MCU designed to support computation intensive application, like ADAS RADAR.

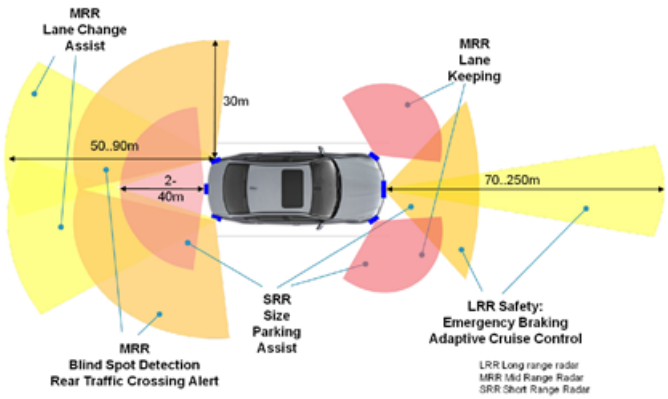

Figure 1. RADAR application overview

2 Signal Processing Toolbox (SPT) in S32R family

In radar signal processing or acceleration, Signal Processing Toolbox (SPT) and Z7 SIMD units are used.

- Accelerated hardware FFT, vector math, peak search, statistics
- Generic DSP functionality with SPE and VFPU unit of Z7 cores

Signal Processing Toolbox (SPT) contains all the hardware modules required for processing of the sampled RADAR signals. It is a powerful processing engine containing high-performance signal processing operations, driven by a specific use-oriented instruction set. The programmability ensures flexibility for modifications of signal processing. The CPU is removed from frequent scheduling of hardware operations, but still controls and interacts with the processing flow.

SPT is connected to the device by an advanced high performance master bus and a peripheral bus.

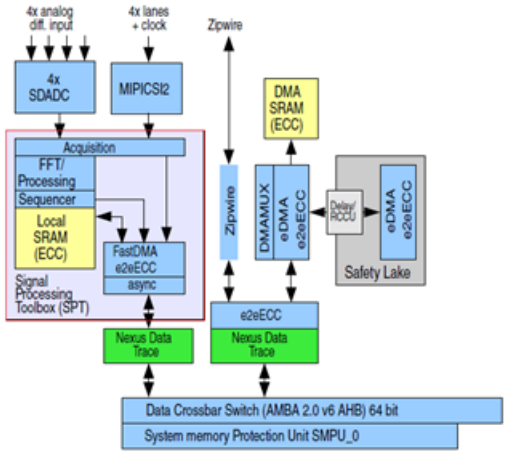

Figure 2. SPT in S32R274 block diagram

The system bus master interface performs fast data transfers between external memory and local RAM.

The purpose of the peripheral interface is to set configuration, get status information and basic control of SPT (start/stop, program pointer) and to trigger interrupts. It can also be used to exchange small amounts of the data between the CPU and the SPT, such as constant operands.

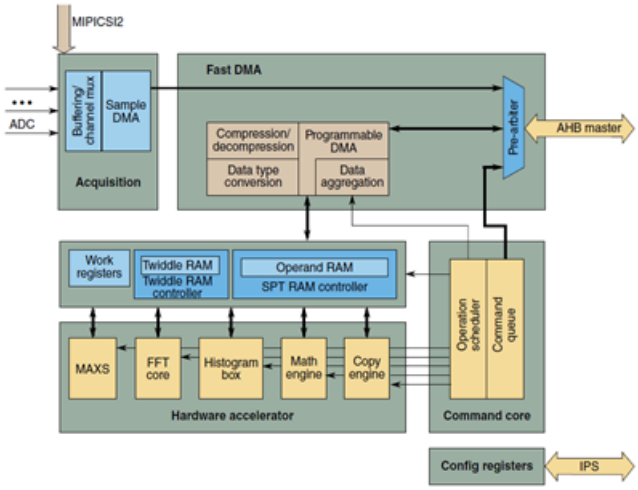

Figure 3. SPT block diagram

It has the following features:

- Acquisition of ADC samples
  - Capturing of ADC samples within programmed window
  - On-the-fly statistical computation
  - Supports MIPICSI2 interface
- Provides HW acceleration for
  - FFT (8 - 4096 point)
  - Histogram calculation
  - Maximum and peak search

  - Mathematical operations on vector data
- High-speed DMA data transfer
  - Supports system RAM, TCM and Flash memory transfers
  - Includes compression/decompression capability for reduction of storage footprint
- Instruction based program flow
  - High-level commands for signal processing operations
  - Simple control commands
  - Local instruction buffer
  - Automatic instruction fetch from main memory
  - CPU interaction possible
- CPU interruption notification
- Watchdog
- Debug Support - Single Stepping and Jamming Mode

It has three operation modes:

1. STOP mode
2. System Debug mode
3. Normal mode

3 SPT execution

The SPT executes a list of instructions. These commands contain all the information to perform signal processing operations on data vectors consisting of a set of numbers. The instruction list is provided by the CPU in a memory buffer (SRAM or flash) and read by the SPT with autonomous triggered DMA operations. Preparation of command scripts is performed off-line during development on a different system, such as a PC.

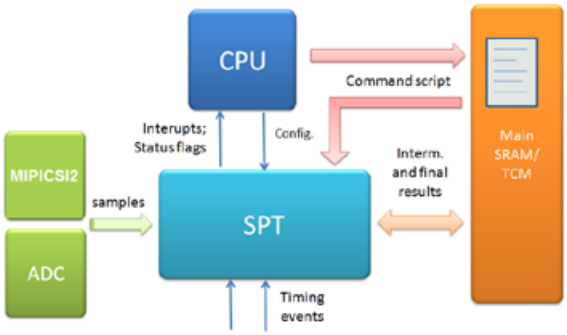

Figure 4. SPT operation model

Data as results or inputs of signal processing operations can be transferred between operand/twiddle RAM and system RAM or TCM by DMA operations. These kind of DMA operations are also scheduled by commands in the script.

CPU application software functions may be invoked between signal processing operations. In order to synchronize these SW functions with the processing flow, interrupts and polling flags are provided, which may be activated as a result of specific commands. In the same way, eDMA operations can be triggered from the SPT. The data transfer descriptors need to be prepared by application software.

To enable advanced and precisely timed pipelining of HW assisted operations, SPT command execution can also be synchronized with or triggered by events from CTE or MIPICSI2.

Program execution is performed by a dedicated register machine. The program execution may be stalled at the end of each command and can be synchronized with events provided by the CTE or MIPICSI2. The CPU can stop and release program execution and execute its own instructions in the same time.

The SPT command sequence is provided by CPU SW and written to a buffer in the system RAM or flash, where CSDMA fetches or alternatively transfers the list of instructions to the command queue.

The program is executed sequentially, one command following the next in the chain with the exception being loops and asynchronous PDMA instruction. The end of a sequence is marked with a specific termination command. The start address of the instruction sequence is configured by application software using the peripheral interface.

The command queue stores a number of instructions to be executed, which may be a shorter sequence than the complete instruction list. It maintains an instruction pointer to the current executed operation, which is incremented when the current operation is completed. This instruction pointer indicates the position in the complete instruction list sequence relative to the first instruction.

Loop instructions are special cases, the program sequence may continue with a lower address depending on the state of the loop counter. The start and next command address of the loops are saved in special registers. Up to four nested loops are supported.

Jump instructions or branch instructions are another case where the program may cause the program to branch to a non-contiguous address location. It is the responsibility of the assembler to ensure that the sanity of the program structure is not violated i.e. infinite loops, jumping into or out of a loop, etc.

4 SPT internal memory

SPT contains internal memory resources namely, Operand RAM, Twiddle RAM and work registers. Out of these, only the work registers can be directly accessed via the peripheral interface and are visible on the SPT external memory map.

The CPU can access these internal memory resources via the hardware accelerators. The instruction bitfields SRC_ADD and DEST_ADD point to these memories.

The Twiddle RAM extends from location 0x4000 to 0x4FFF. However the area from 0x5000 to 0x7FFF is aliased to the Twiddle RAM i.e. it wraps around to point to the Twiddle RAM. It should be noted that the memory accesses to the Twiddle RAM should not cross the area from 0x4000 to 0x7FFF.

The Operand RAM extends from location 0x8000 to 0xBFFF. But the area from 0xC000 to 0xFFFF is aliased to the Operand RAM. The memory accesses to the Operand RAM should not cross the area from 0x8000 to 0xFFFF.

5 Common instruction fields

Common instructions contains the following fields.

- Opcode
- Source and Destination address
- Indirect Memory address
- Source and destination address increment
- Vector size
- Input datatype/preprocessing

6 SPT programming

6.1 Programming based on RSDK

NXP Radar SDK(RSDK) provides basic radar processing algorithms and device drivers for S32R hardware devices. Its purpose is to facilitate radar algorithm development (using SPT kernels, MATLAB models), creation of higher level algorithms (starting from the basic blocks supplied with RSDK) and easy application development by integrating driver and platform support.

Radar SDK (Radar SDK for S32R27 SPT accelerator) could be downloaded from NXP website.

RSDK SPT module consists of the SPT Driver, SPT Kernels software components.

The SPT Driver serves as an interface between the user application running on the host CPU cores and the SPT hardware. The figure below shows a high-level block diagram of SPT software architecture. Its purpose is to enable the integration and execution of low-level microcode kernels on the SPT (Signal Processing Toolbox) accelerator for baseband radar signal processing.

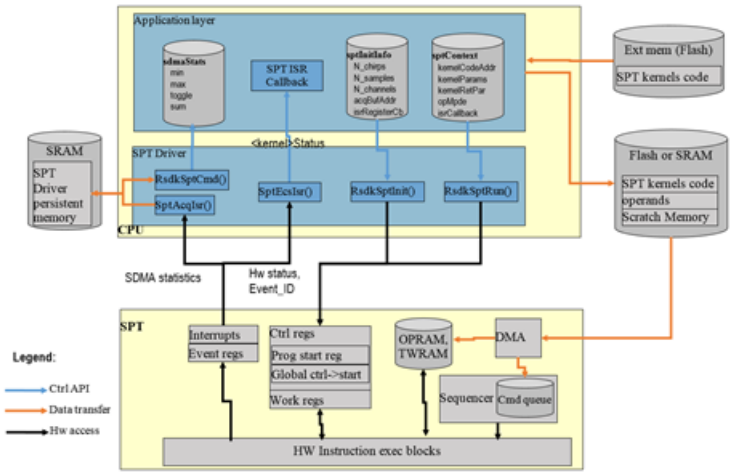

Figure 5. SPT software architecture

The SPT Kernels are individual precompiled routines of SPT code packed into a library, each implementing a part of the radar processing flow.

Together, the SPT Driver and SPT Kernels library provide a software abstraction of the built-in SPT hardware functions (e.g. FFT, Maximum Search, Histogram etc.) combined in a series of algorithms.

The RSDK package also includes RSDK Sample Applications showing how to integrate these software components into the user code.

6.2 Programming based on graphic tool

S32 Design Studio SPT graphical tool allows user to use graphical modeling workbench by leveraging the Eclipse Modeling technologies. It provides a workbench for model-based architecture engineering. Graphical tool equips teams who have to deal with complex architectures. The graphical tool includes everything necessary to easily create and manipulate models. The output of this tool is SPT assembler source code.

The SPT graphical tool is integrated with the New S32DS Project wizard for devices which have SPT module. SPT1, SPT2 and SPT2.5 versions of SPT modules are supported. The SPT graphical tool can be used after project creation with the new graph tools project wizard. For detailed description of new project wizards, please read S32 Design Studio for Power Architecture, Version 2017.R1 reference manual.

After the project is generated, SPT graphical tool windows could be gotten in IDE. User could drag 'Instructions', 'Flow' and 'Directives' from Palette window into working flow window.

Each 'Instructions', 'Flow' or 'Directives' could change their setting in properties window. After finished the design of SPT working flow, generate SPT code through right click mouse.

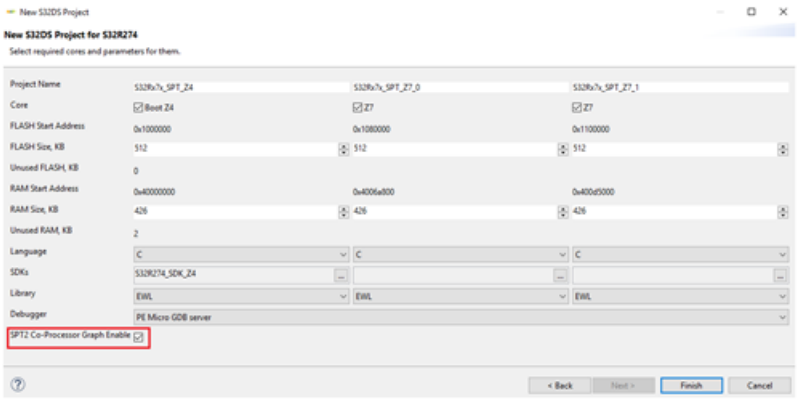

Figure 6. Generate project with SPT graphical tool

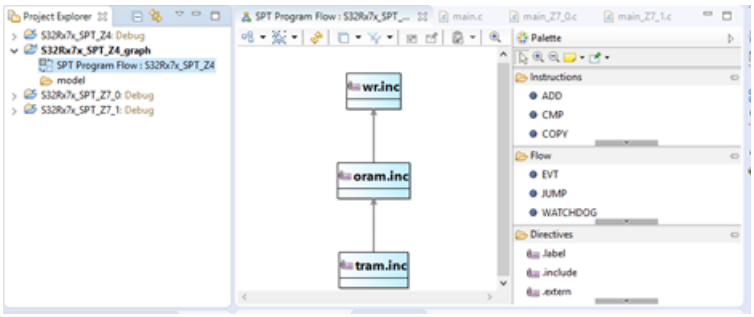

Figure 7. SPT graphical tool windows

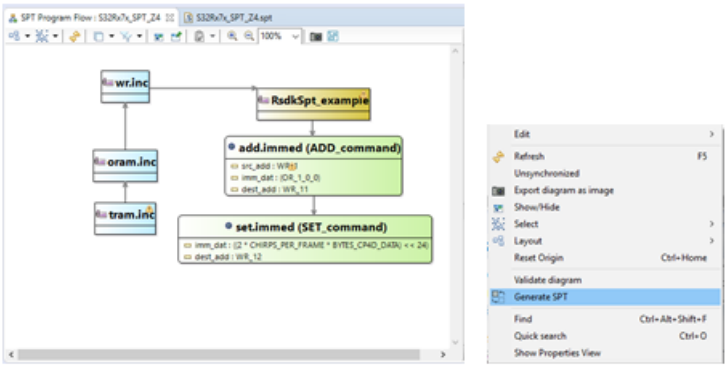

Figure 8. Example working flow and generate code

7 Conclusion

The SPT is a powerful processing engine containing high-performance signal processing operations driven by a specific instruction set. Its programmability ensures flexibility while removing the CPU from frequent scheduling of hardware operations, while still controlling and interacting with the processing flow. RSDK supports the API of SPT for customer and graphics tool for SPT programming could meet the flexible design request.

8 Reference

- S32R274 Reference Manual
- Radar SDK for S32R27 SPT accelerator
- AN5375, S32R RADAR Signal Compression

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: May 2019
Document identifier: AN12424