Introduction to programmable logic array (PLAY) for STM32 MCUs

Introduction

The programmable logic array (PLAY) allows users to design custom logic circuits and state machines without needing external devices like FPGAs. It uses look-up tables with multiple inputs to create flexible logical functions. Inputs and outputs can be selected from various sources, and the system supports synchronization, filtering, and feedback for complex designs. Configuration is simple, protected, and persistent through resets.

Table 1. Applicable products

Type	Products
Microcontrollers	STM32H5E4/5F4 line STM32H5E5/5F5 line

1 General information

This document applies to the Arm® Cortex® core-based STM32 microcontrollers.

Reference documents

[1] PLAY wiki - Getting started with Programmable Logic ArraY (PLAY)

2 Overview

2.1 Features

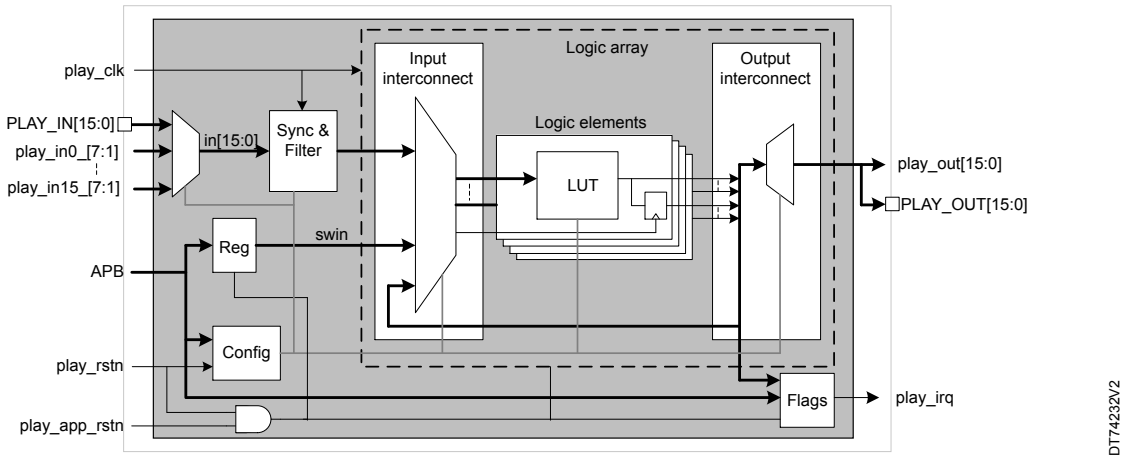
- Glitch-free programmable logic elements each comprising a 4-input look-up table and a register
- Configurable input and output interconnects
- Optional synchronization of inputs, with programmable glitch filters and edge
- Detection/pulse extension feature
- Software programmable inputs
- Flags with interrupt capability for communication with software
- Simple register-based configuration interface, protected by software lock
- Arm® TrustZone®-aware
- Privilege-aware

Note: Arm and TrustZone are registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere.

2.2 Implementation

Figure 1 shows the block diagram for the programmable logic array PLAY.

Figure 1. PLAY block diagram



2.2.1 Input

Inputs to the logic array can be chosen from external I/Os, internal signals, or software-programmable register bits. External inputs may be synchronized with the internal clock and filtered to eliminate short pulses, or they can be connected directly to the logic array for purely combinational or asynchronous functions.

Inputs can be:

- Internal: play_in[15:0][7:1]
- External: PLAY_IN[15:0]
- Software input: swin[15:0]

Each input, except swin[15:0], passes optionally through a filter, which can remove glitches or short pulses and generate a pulse if an edge is detected.

A pre-multiplexer selects which signal is connected to an input. The pre-multiplexer for logic array input n is configured by the PLAY_FILTnCFG.PREMUXSEL[2:0] register field. Only one signal can be chosen per input.

Note: For STM32H5Ex/5Fx, 128 input sources: 16 from I/Os and 112 internal signals.

2.2.2 Output

The PLAY output includes the following signals:

- PLAY_OUT [15:0]: External output from the PLAY (can be connected to GPIOs)
- play_out [15:0]: Internal output (connected to other functional blocks)

For more details about the PLAY output connections, refer to the device reference manual.

2.2.3 Logic elements

The logic array provides 16 logic elements (LE's), each containing:

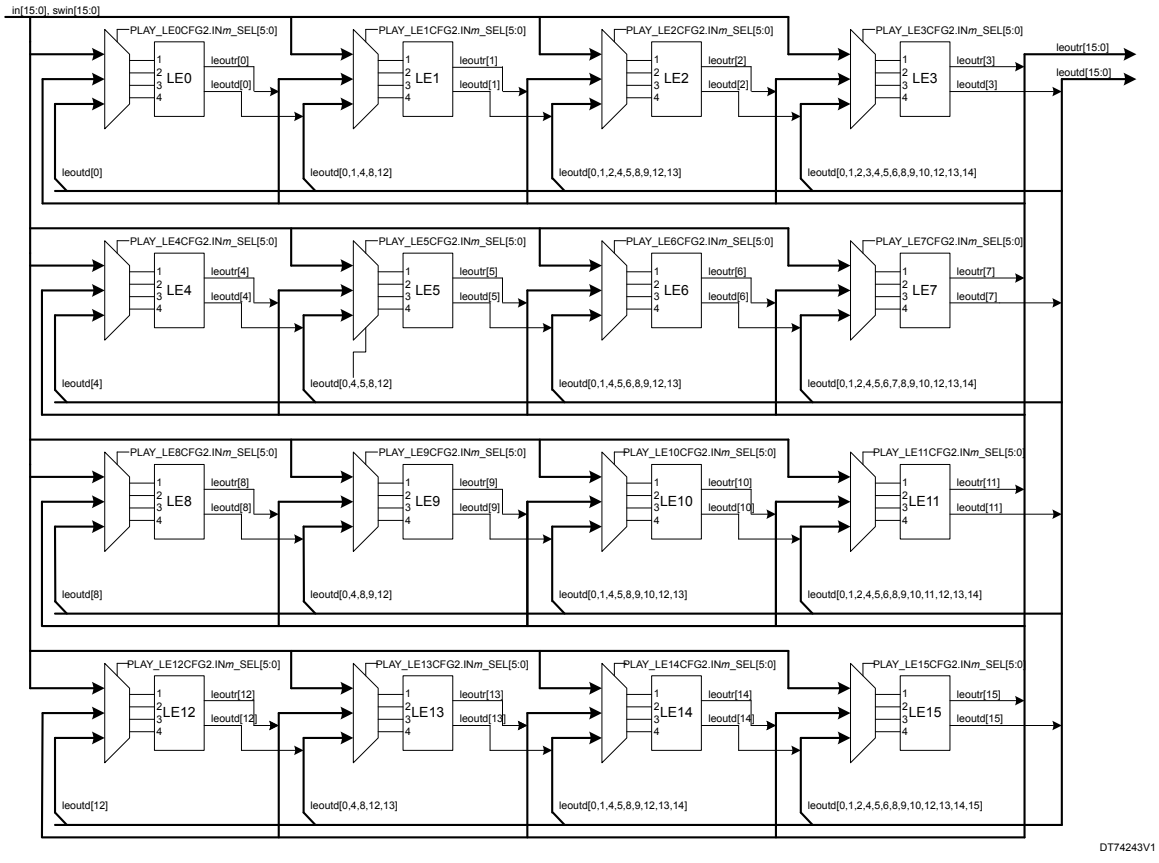
- Look-up table (LUT)
- Direct output
- Registered output (clocked by the play_clk, optionally qualified by a clock enable signal)

Each LE has up to 4 inputs that can be selected as LUT inputs from:

- External/internal inputs (in[15:0])
- Software programmable inputs (swin[15:0])
- Feedback from registered LE outputs (leoutr[15:0])
- Constant logic 0

It is also possible to connect the direct outputs of some logic elements to the inputs of others; see Figure 2.

Figure 2. Input interconnect



Block 1: LE 0, 4, 8, and 12

Each LE can use its own direct output as input.

Block 2: LE 1, 5, 9, and 13

Each LE can use its own direct output as input and the direct outputs of the LEs in block 1.

Block 3: LE 2, 6, 10, and 14

Each LE can use its own direct output as input and the direct outputs of the LEs in block 1 and block 2.

Block 4: LE 3, 7, 11, and 15

Each LE can use its own direct output as input and the direct outputs of the LEs in blocks 1 to 3.

3 Ecosystem

This section describes the software ecosystem used in this application note, from early logic design to STM32 firmware implementation. The proposed flow relies on three main elements:

- LOGISIM: open source tool for optional logic design and simulation
- STM32CubeMX for STM32 configuration and code generation
- Integrated Development Environments (IDEs) for code editing, build and debug

Together, these tools provide a complete and consistent path from a high-level logic concept to a fully functional STM32-based implementation.

Note: LOGISIM is not mandatory: if the behavior of the custom application is already well defined and validated, the design can start directly from STM32CubeMX.

3.1 LOGISIM: digital logic design and simulation (optional)

LOGISIM is used as the entry point of the design flow to model and validate the digital logic before targeting the STM32 microcontroller.

Key features are:

- Graphical circuit editor
  LOGISIM provides an intuitive graphical interface to place logic gates, flip-flops, multiplexers, and other digital components, and interconnect them to form the target circuit.
- Truth-table based design
  The embedded combinational analysis module allows the designer to create circuits directly from truth tables. This is particularly useful to derive optimal logic equations and verify functional behavior.
- Interactive simulation
  Input stimuli can be applied in real time, while outputs and internal nodes are monitored to ensure the design meets the intended specification. This reduces the risk of functional issues during firmware implementation.
- Cross-platform availability
  LOGISIM runs under Linux, macOS, and Windows, making it easily accessible in heterogeneous development environments.

In the context of this application note, LOGISIM is used to:

- Define the target digital behavior at block level.
- Validate the functional correctness of the logic using simulation.

When the behavior of the custom application is already well specified and validated, the user can skip the LOGISIM step and start directly with STM32CubeMX configuration and firmware development.

3.2 STM32CubeMX: STM32 configuration and code generation

Once the logic is validated with LOGISIM, or when the custom application is already clearly defined, map the design to the STM32 microcontroller using STM32CubeMX. STM32CubeMX is the tool provided by STMicroelectronics to generate code for configuration and initialization of STM32 devices.

- Generate initialization code
  STM32CubeMX automatically generates C project files, including:
  - System initialization and clock setup
  - HAL, LL drivers, and basic middleware as required

The complete ecosystem provides a structured and flexible workflow:

- Optionally perform functional design and validation in LOGISIM when the logic behavior still needs to be explored or demonstrated.
- Generate STM32 configuration and initialization code using STM32CubeMX.
- Develop, build, and debug application firmware in the chosen integrated development environment (IDE).

If the custom application is already well defined and its logic is known, the designer can bypass LOGISIM and start directly from the STM32CubeMX configuration step.

4 PLAY examples for STM32H5Ex/5Fx

4.1 Available examples for STM32H5Ex/5Fx

The STM32CubeH5 firmware package includes several programmable logic array (PLAY) example projects to help users get started with this feature on STM32H5 devices. These examples demonstrate how to configure the PLAY block with STM32CubeMX and how to interface custom logic with on-chip peripherals, such as GPIO, timer, and others.

The projects are available in the standard Projects directory of the STM32CubeH5 package. The projects cover typical use cases, including:

- **PLAY_Counter_3bits** – Shows how to configure the PLAY peripheral to implement a 3-bit counter using interconnected Lookup Tables (LUTs) and the PLAY input signal.
- **PLAY_Emergency_Alarm** – Demonstrates how to configure PLAY to implement a more complex logic with multiple interconnected LUTs. The example implements an emergency alarm system that outputs a Morse SOS signal without using the CPU, even in Stop mode.
- **PLAY_Gate_XOR** – Illustrates how to configure the PLAY peripheral to act as a XOR gate. The example performs a logical XOR operation on two input signals using PLAY.
- **PLAY_Signal_Routing** – Shows how to configure the PLAY peripheral to act as a signal router. In this example, PLAY routes a timer internal signal (TIM) to an external GPIO pin.

4.2 PLAY wiki

The PLAY wiki offers a practical step-by-step guide for using the PLAY peripheral on STM32H5 devices. It complements the STM32CubeH5 firmware package by explaining how to design, simulate, and deploy custom logic functions and small state machines using PLAY, without adding external logic components or significantly loading the CPU.

The PLAY wiki walks through several hands-on examples—such as a simple XOR gate, a 3-bit counter, and signal routing between internal peripherals and GPIOs—starting from logic design in Logisim and ending with configuration in STM32CubeH5.

For detailed procedures, screenshots, and example projects, refer to the PLAY wiki.

Revision history

Table 2. **Document revision history**

Date	Version	Changes
05-Mar-2026	1	Initial release.

Contents

List of tables

List of figures