```c
/*
 * dma_driver.h
 *
 *  Created on: Dec 11, 2018
 *      Author: Dominic Doty
 */

#ifndef DMA_DRIVER_H_
#define DMA_DRIVER_H_

/* INCLUDES */
#include "MKL25Z4.h"
#include "stddef.h"
#include "fsl_common.h"
#include "fsl_clock.h"


/* DEFINES & TYPEDEFS */

// DMA Errors
typedef enum
{
    DMA_ERROR_SUCCESS,
    DMA_ERROR_NULL_PTR,
    DMA_ERROR_BAD_ADDR,
    DMA_ERROR_BUSY,
    DMA_ERROR_BYTE_COUNT,
    DMA_ERROR_UNKNOWN_DMA
} dma_error;

// DMA Channels
typedef enum
{
    DMA_CHANNEL_0,
    DMA_CHANNEL_1,
    DMA_CHANNEL_2,
    DMA_CHANNEL_3
} dma_channel;

// DMA Data Sizes
typedef enum
{
    DMA_SIZE_32,
    DMA_SIZE_8,
    DMA_SIZE_16
} dma_size;

// DMA Modulos
typedef enum
{
    DMA_MOD_NONE,
    DMA_MOD_16b,
    DMA_MOD_32b,
    DMA_MOD_64b,
    DMA_MOD_128b,
    DMA_MOD_256b,
    DMA_MOD_512b,
    DMA_MOD_1k,
    DMA_MOD_2k,
    DMA_MOD_4k,
    DMA_MOD_8k,
    DMA_MOD_16k,
    DMA_MOD_32k,
    DMA_MOD_64k,
    DMA_MOD_128k,
    DMA_MOD_256k
} dma_mod;

// DMA Link Channel Mode
```

```c
typedef enum
{
    DMA_LINK_NONE,
    DMA_LINK_LCH1_ON_CS_LCH2_AND_BCR_ZERO,
    DMA_LINK_LCH1_ON_CS,
    DMA_LINK_LCH1_ON_BCR_ZERO
} dma_link_mode;

// DMA Link Channels
typedef enum
{
    DMA_LINK_DMA_CHAN_0,
    DMA_LINK_DMA_CHAN_1,
    DMA_LINK_DMA_CHAN_2,
    DMA_LINK_DMA_CHAN_3
} dma_link_channel;

// DMA Mux Configuration
typedef struct
{
    DMAMUX_Type* dma_mux;
    dma_channel channel;
    bool channel_enable;
    bool trigger_mode;
    dma_request_source_t slot;
} dma_mux_config;

#define DMA_MUX_CONFIG_DEFAULT        \
{                                    \
    .dma_mux = DMAMUX0,                \
    .channel = DMA_CHANNEL_0,        \
    .channel_enable = false,        \
    .trigger_mode = false,            \
    .slot = kDmaRequestMux0ADC0        \
}

// DMA Configuration
typedef struct
{
    DMA_Type* dma;
    dma_channel channel;
    volatile void* src_addr;
    volatile void* dest_addr;
    uint32_t byte_count;
    bool interrupt;
    bool peripheral_en;
    bool steal_cycles;
    bool auto_align;
    bool async_en;
    bool src_inc;
    dma_size src_size;
    dma_mod src_mod;
    bool dest_inc;
    dma_size dest_size;
    dma_mod dest_mod;
    bool auto_disable_req;
    dma_link_mode link_mode;
    dma_link_channel link_chan_1;
    dma_link_channel link_chan_2;
    bool start;
} dma_init_config;

#define DMA_INIT_CONFIG_DEFAULT            \
{                                        \
    .dma = NULL,                        \
    .channel = DMA_CHANNEL_0,            \
    .src_addr = NULL,                    \
    .dest_addr = NULL,                    \
    .byte_count = 0,                    \
    .interrupt = false,                    \
    .peripheral_en = false,                 \
    .steal_cycles = false,                  \
    .auto_align = false,                    \
    .async_en = false,                      \
    .src_inc = false,                       \
    .src_size = DMA_SIZE_32,                \
    .src_mod = DMA_MOD_NONE,                \
    .dest_inc = false,                      \
    .dest_size = DMA_SIZE_32,               \
    .dest_mod = DMA_MOD_NONE,               \
    .auto_disable_req = false,              \
    .link_mode = DMA_LINK_NONE,             \
    .link_chan_1 = DMA_LINK_DMA_CHAN_0,     \
    .link_chan_2 = DMA_LINK_DMA_CHAN_0,     \
    .start = false                          \
}

/* FUNCTION DECLARATIONS */

// DMA Initialization
dma_error dma_init(dma_init_config* config);

// DMA Mux Initialization
dma_error dma_mux_init(dma_mux_config* config);

// Enable or Disable a Mux Channel
void dma_mux_channel_enable(DMAMUX_Type* dma_mux, dma_channel channel, bool enable);

// Used to restart a DMA transfer on an already configured DMA Channel (resets peripheral_en)
void dma_transfer_restart(DMA_Type* dma, dma_channel channel, volatile void* buffer_ptr, uint32_t byte_count);

#endif /* DMA_DRIVER_H_ */
```