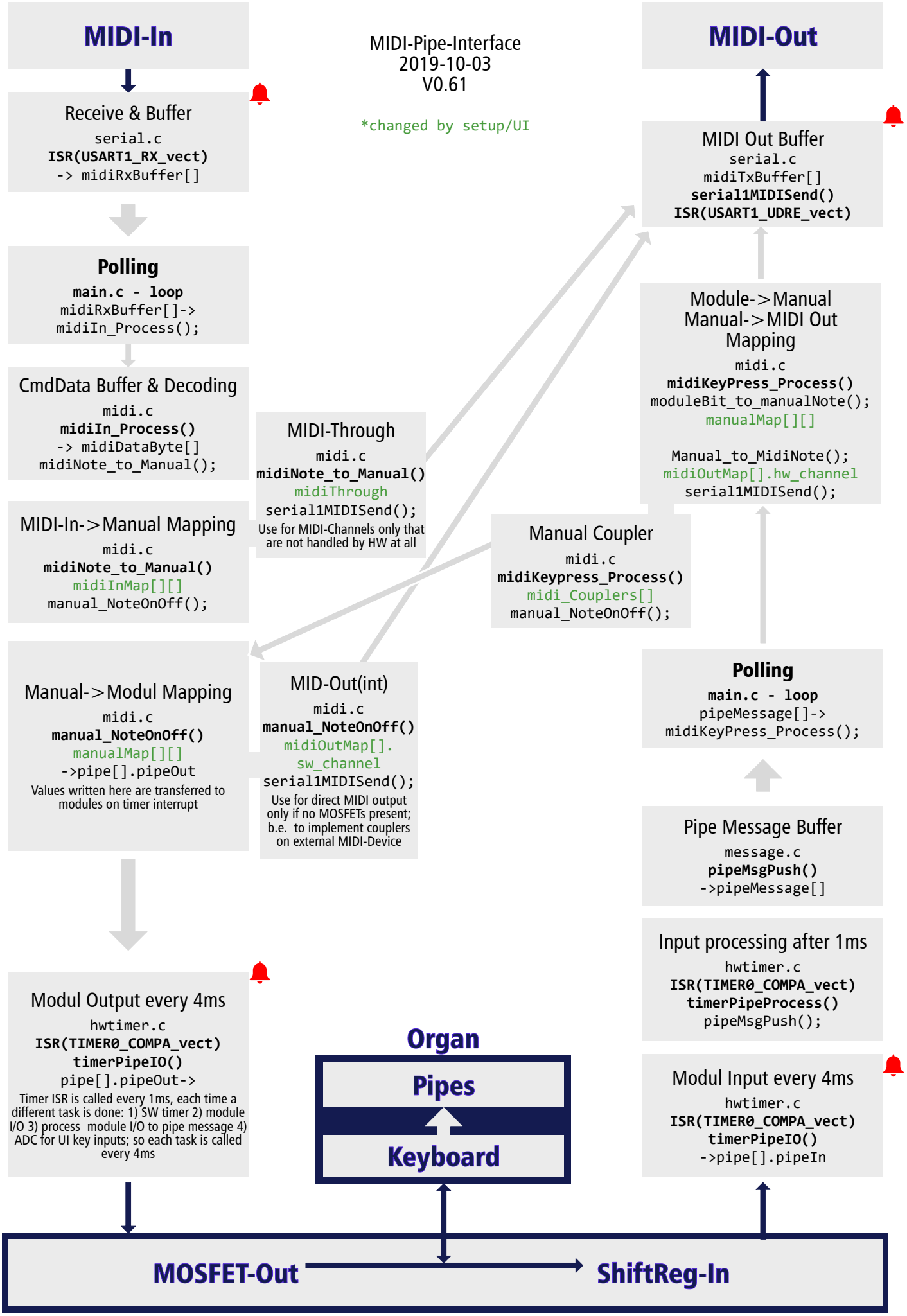
MIDI-In
MIDI-Pipe-Interface
2019-10-03
V0.61
*changed by setup/UI
MIDI-Out
Receive & Buffer
serial.c
ISR(USART1_RX_vect)
-> midiRxBuffer[]
MIDI Out Buffer
serial.c
midiTxBuffer[]
serial1MIDISend()
ISR(USART1_UDRE_vect)
Polling
main.c - loop
midiRxBuffer[]->
midiIn_Process();
Module->Manual
Manual->MIDI Out
Mapping
midi.c
midiKeyPress_Process()
moduleBit_to_manualNote();
manualMap[][]
Manual_to_MidiNote();
midiOutMap[].hw_channel
serial1MIDISend();
CmdData Buffer & Decoding
midi.c
midiIn_Process()
-> midiDataByte[]
midiNote_to_Manual();
MIDI-Through
midi.c
midiNote_to_Manual()
midiThrough
serial1MIDISend();
Use for MIDI-Channels only that are not handled by HW at all
MIDI-In->Manual Mapping
midi.c
midiNote_to_Manual()
midiInMap[][]
manual_NoteOnOff();
Manual Coupler
midi.c
midiKeypress_Process()
midi_Couplers[]
manual_NoteOnOff();
Manual->Modul Mapping
midi.c
manual_NoteOnOff()
manualMap[][]
->pipe[].pipeOut
Values written here are transferred to modules on timer interrupt
MID-Out(int)
midi.c
manual_NoteOnOff()
midiOutMap[].
sw_channel
serial1MIDISend();
Use for direct MIDI output only if no MOSFETs present; b.e. to implement couplers on external MIDI-Device
Polling
main.c - loop
pipeMessage[]->
midiKeyPress_Process();
Pipe Message Buffer
message.c
pipeMsgPush()
->pipeMessage[]
Input processing after 1ms
hwtimer.c
ISR(TIMER0_COMPA_vect)
timerPipeProcess()
pipeMsgPush();
Modul Output every 4ms
hwtimer.c
ISR(TIMER0_COMPA_vect)
timerPipeIO()
pipe[].pipeOut->
Timer ISR is called every 1ms, each time a different task is done: 1) SW timer 2) module I/O 3) process module I/O to pipe message 4) ADC for UI key inputs; so each task is called every 4ms
Organ
Pipes
Keyboard
Modul Input every 4ms
hwtimer.c
ISR(TIMER0_COMPA_vect)
timerPipeIO()
->pipe[].pipeIn
MOSFET-Out
ShiftReg-In
Module
Note: Due to HW design MOSFET activitation is treated as key press in SW, so processing continues here to right upper side