Digital Signal Processing for Music
Part 17: real-time and blocking

alexander lerch

Georgia Tech | Center for Music Technology
College of Design
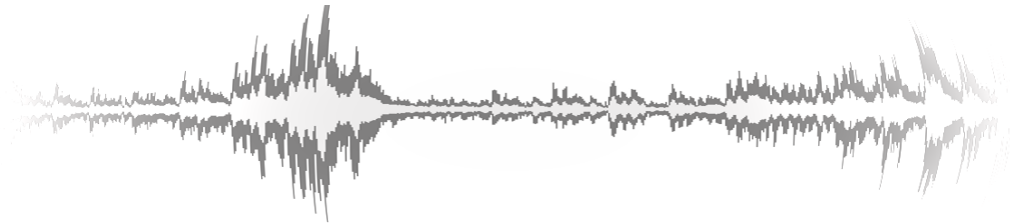
real-time systems
introduction

- many audio processing systems are real-time systems
- this includes
  - most audio plugins,
  - studio hardware effects etc.

real-time systems
introduction

Georgia Tech | Center for Music Technology
College of Design

real-time system (wikipedia)

"In a real-time digital signal processing (DSP) process, the analyzed (input) and generated (output) samples can be processed (or generated) continuously in the time it takes to input and output the same set of samples independent of the processing delay"

- "processing delay and resources must be bounded even if the processing continues for an unlimited time"
- "mean processing time per sample is no greater than the sampling period, which is the reciprocal of the sampling rate"
- ⇒ "perform all computations continuously at a fast enough rate that the output (…) keeps up with changes in the input signal"

real-time systems
introduction

real-time system (wikipedia)

"In a real-time digital signal processing (DSP) process, the analyzed (input) and generated (output) samples can be processed (or generated) continuously in the time it takes to input and output the same set of samples independent of the processing delay"

- "processing delay and resources must be bounded even if the processing continues for an unlimited time"
- "mean processing time per sample is no greater than the sampling period, which is the reciprocal of the sampling rate"
- ⇒ "perform all computations continuously at a fast enough rate that the output (…) keeps up with changes in the input signal"

real-time systems
introduction

real-time system (wikipedia)

"In a real-time digital signal processing (DSP) process, the analyzed (input) and generated (output) samples can be processed (or generated) continuously in the time it takes to input and output the same set of samples independent of the processing delay"

- "processing delay and resources must be bounded even if the processing continues for an unlimited time"

- "mean processing time per sample is no greater than the sampling period, which is the reciprocal of the sampling rate"

⇒ "perform all computations continuously at a fast enough rate that the output (…) keeps up with changes in the input signal"

real-time systems
introduction

Georgia Tech | **Center for Music Technology**
College of Design

> **real-time system** (wikipedia)
>
> "In a real-time digital signal processing (DSP) process, the analyzed (input) and generated (output) samples can be processed (or generated) continuously in the time it takes to input and output the same set of samples independent of the processing delay"

- "processing delay and resources must be bounded even if the processing continues for an unlimited time"
- "mean processing time per sample is no greater than the sampling period, which is the reciprocal of the sampling rate"
- ⇒ "perform all computations continuously at a fast enough rate that the output (...) keeps up with changes in the input signal"

real-time systems
properties

- **performance**:
    - processing time for one block $\leq$ block length
    - real-time computing does not necessarily mean high performance computing!

- causality:
    - system output/state depends only on current and prior values
    - *no* knowledge of future samples

- latency:
    - delay of a system between the stimulus and the response to this stimulus
        - *algorithmic delay*: (FFT-Processing, Look-Ahead, . . .)
        - *interface delay*: (block length, ad/da conversion)

real-time systems
properties

- **performance**:
  - processing time for one block $\leq$ block length
  - real-time computing does not necessarily mean high performance computing!

- causality:
  - system output/state depends only on current and prior values
  - *no* knowledge of future samples

- latency:
  - delay of a system between the stimulus and the response to this stimulus
    - *algorithmic delay*: (FFT-Processing, Look-Ahead, . . .)
    - *interface delay*: (block length, ad/da conversion)

real-time systems
properties

- **performance**:
  - processing time for one block $\leq$ block length
  - real-time computing does not necessarily mean high performance computing!

- **causality**:
  - system output/state depends only on current and prior values
  - *no* knowledge of future samples

- latency:
  - delay of a system between the stimulus and the response to this stimulus
    - *algorithmic delay*: (FFT-Processing, Look-Ahead, . . .)
    - *interface delay*: (block length, ad/da conversion)
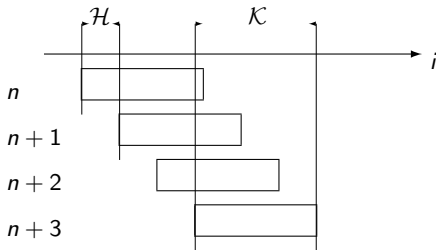
real-time systems
properties

- **performance**:
  - processing time for one block $\leq$ block length
  - real-time computing does not necessarily mean high performance computing!

- **causality**:
  - system output/state depends only on current and prior values
  - *no* knowledge of future samples

- **latency**:
  - delay of a system between the stimulus and the response to this stimulus
    - *algorithmic delay*: (FFT-Processing, Look-Ahead, . . .)
    - *interface delay*: (block length, ad/da conversion)

real-time systems
block based processing
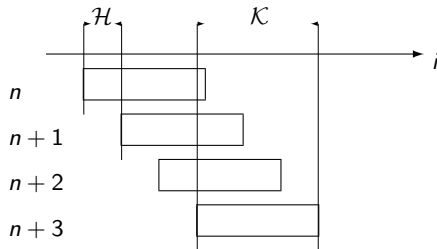
processing of *blocks of samples* vs. individual samples

reasons:
- block based algorithms (FFT, . . .)
- audio hardware characteristics
- efficiency (SIMD, memory allocation)

real-time systems
block based processing

processing of *blocks of samples* vs. individual samples

reasons:

- block based algorithms (FFT, . . .)
- audio hardware characteristics
- efficiency (SIMD, memory allocation)

real-time systems
block sizes

- typical block sizes can range from 1... thousands of samples
- often powers of 2

- in many DAWs and some drivers the **block size varies**

real-time systems
block sizes

- typical block sizes can range from 1...thousands of samples
- often powers of 2

- in many DAWs and some drivers the **block size varies**

real-time systems
time stretching and pitch shifting

can pitch shifting theoretically be implemented as real-time system

real-time systems
time stretching and pitch shifting

can pitch shifting theoretically be implemented as real-time system

Yup.

can time stretching theoretically be implemented as real-time system

real-time systems
time stretching and pitch shifting

can pitch shifting theoretically be implemented as real-time system

Yup.

can time stretching theoretically be implemented as real-time system

Nope. Explain.

real-time systems
inplace processing

Georgia Tech | Center for Music Technology
College of Design

what is "inplace processing"

real-time systems
inplace processing

what is "inplace processing"

- samples of the input block are replaced with the output block

real-time systems
inplace processing

Georgia Tech | Center for Music Technology
College of Design

what is "inplace processing"

- samples of the input block are replaced with the output block
  - + resource friendly: memory allocation for output buffer
  - − original input data cannot be used anymore

real-time systems
blocking

- **time-stamps**
  - blocking can be considered similar to down-sampling
  - ⇒ *what time stamps to assign to each block?*
    - begin of each block
    - center of each block
- **initialization**
  - real-time systems are designed to work for infinite input stream
  - ⇒ *how to initialize internal buffers?*
    - usually zeros, but other initializations may make sense in specific scenarios
- **performance** issues due to blocking
  - plugin gets stream of samples split into small blocks (e.g., 32 samples)
  - internally, STFT with large hopsize (e.g., 2048 samples) is used
  - ⇒ *what is the potential performance problem here?*
    - each hop requires data from 64 input blocks
    - no processing can be done for 63 blocks
    - processing of huge FFT has to be done during the 64th block (32 samples)

real-time systems
blocking

Georgia | Center for Music
Tech | Technology
College of Design

- **time-stamps**
    - blocking can be considered similar to down-sampling
  ⇒ *what time stamps to assign to each block?*
        - begin of each block
        - center of each block

- **initialization**
    - real-time systems are designed to work for infinite input stream
  ⇒ *how to initialize internal buffers?*
        - usually zeros, but other initializations may make sense in specific scenarios

- **performance** issues due to blocking
    - plugin gets stream of samples split into small blocks (e.g., 32 samples)
    - internally, STFT with large hopsize (e.g., 2048 samples) is used
  ⇒ *what is the potential performance problem here?*
        - each hop requires data from 64 input blocks
        - no processing can be done for 63 blocks
        - processing of huge FFT has to be done during the 64th block (32 samples)

real-time systems
blocking

- **time-stamps**
  - blocking can be considered similar to down-sampling
  - ⇒ *what time stamps to assign to each block?*
    - begin of each block
    - center of each block

- **initialization**
  - real-time systems are designed to work for infinite input stream
  - ⇒ *how to initialize internal buffers?*
    - usually zeros, but other initializations may make sense in specific scenarios

- **performance** issues due to blocking
  - plugin gets stream of samples split into small blocks (e.g., 32 samples)
  - internally, STFT with large hopsize (e.g., 2048 samples) is used
  - ⇒ *what is the potential performance problem here?*
    - each hop requires data from 64 input blocks
    - ⇒ no processing can be done for 63 blocks
    - ⇒ processing of huge FFT has to be done during the 64th block (32 samples)

real-time systems
blocking

Georgia | Center for Music
Tech | Technology
College of Design

- **time-stamps**
  - blocking can be considered similar to down-sampling
  - ⇒ *what time stamps to assign to each block?*
    - begin of each block
    - center of each block

- **initialization**
  - real-time systems are designed to work for infinite input stream
  - ⇒ *how to initialize internal buffers?*
    - usually zeros, but other initializations may make sense in specific scenarios

- **performance** issues due to blocking
  - plugin gets stream of samples split into small blocks (e.g., 32 samples)
  - internally, STFT with large hopsize (e.g., 2048 samples) is used
  - ⇒ *what is the potential performance problem here?*
    - each hop requires data from 64 input blocks
    - ⇒ no processing can be done for 63 blocks
    - ⇒ processing of huge FFT has to be done during the 64th block (32 samples)

summary
real-time systems

real-time systems have the following properties:

- hard **performance** requirements
  - processing of input block has to be faster then time span of this block **for all blocks, not only on average**

- **causality**
  - future samples cannot taken into account (or only by increasing the latency: look-ahead)

- **latency**
  - time between input and system response, usually intended to be minimal