Ruby - Bug #15033

Encoding fallback uses wrong character when multiple conversions are required

08/27/2018 09:19 PM - stevecheckoway (Stephen Checkoway)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.5.1p57 (2018-03-29 revision 63029) [x86_64-darwin17]	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

When converting a string from one encoding to another that involves multiple conversions, the proc passed to encode will be called with the incorrect value if the conversion fails in the middle of the conversion.

For example,

```
> "\u016f".encode('ISO-2022-JP', fallback: proc { |c| "&\#x#{c.ord.to_s(16)};" }).encode('UTF-8')
=> "&#x8fabeb;"
```

Here, the ordinal passed to the proc was 0x8fabeb rather than 0x16f.

If I use the :xml option instead of :fallback, I get the expected result

```
> "\u016f".encode('ISO-2022-JP', xml: :text).encode('UTF-8')
=> "&#x16F;"
```

The cause of this seems pretty clear. The conversion process from UTF-8 to ISO-2022-JP goes from UTF-8 to EUC-JP to stateless-ISO-2022-JP to ISO-2022-JP. The first conversion succeeds

```
> "\u016f".encode('EUC-JP')
=> "\x{8FABEB}"
```

but the second fails

```
> "\u016f".encode('EUC-JP').encode('stateless-ISO-2022-JP')
Traceback (most recent call last):
        3: from /opt/local/bin/irb2.5:11:in `<main>'
        2: from (irb):10
        1: from (irb):10:in `encode'
Encoding::UndefinedConversionError ("\x8F\xAB\xEB" from EUC-JP to stateless-ISO-2022-JP)
```

In this situation, I believe that the procedure passed to encode should be called with the original failing character, not an intermediate one.

History

#1 - 12/09/2019 09:57 AM - naruse (Yui NARUSE)

- Status changed from Open to Closed

It is feature.
What it does when it's specified xml: :text is just like below:

```
"\u016f".encode('ISO-2022-JP', fallback: proc { |c| "&\#x#{c.encode('UTF-8').ord.to_s(16)};" }).encode('UTF-8'
)
```